YOUR OFFICIAL RASPBERRY PI MAGAZINE

The MagPi

Issue 141 | May 2024 | magpi.cc

The official Raspberry Pi magazine

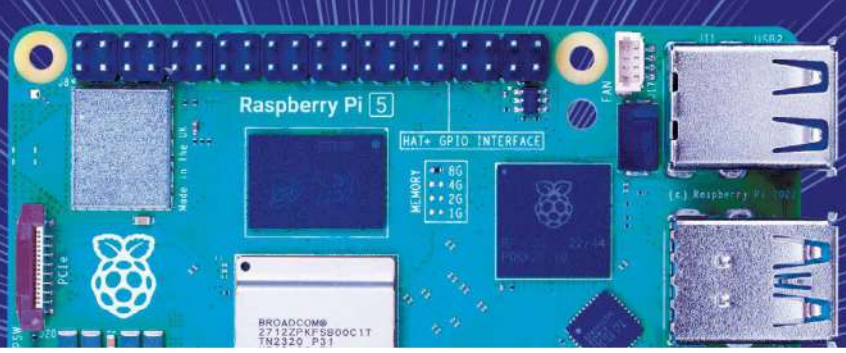
RASPBERRY PI AI MADE CLEAR

ROLL YOUR OWN MODELS

PiDP-10
retro classic

Raspberry Pi 5
cases on test

Image & text generation • Natural voice synthesis • AI robots, cameras & kits



£5.99

9 772051 998001 05

LEARN TO CONTROL AN **INDUSTRIAL ROBOT ARM**

Industrial Raspberry Pi
ComfilePi

UL LISTED

Powered by Raspberry Pi

The ComfilePi is a touch panel PC designed with high-tolerant components and no moving parts for industrial applications. It features a water-resistant front panel, touchscreen, color LCD (available in various sizes), RS-232, RS-485, Ethernet, USB, I2C, SPI, digital IO, battery-backed RTC (real-time clock), and piezo buzzer.

Use the rear-panel 40-pin GPIO header to expand its features and capabilities with additional I/O boards. The ComfilePi is UL Listed and employs Raspberry Pi Compute Module.

Visit www.comfiletech.com

COMFILE TECHNOLOGY

WELCOME
to The MagPi 141

EDITOR
Lucy Hattersley

Lucy is still the editor of *The MagPi* until she can train a bot to do it!

magpi.cc

Artificial Intelligence remains the hottest topic in technology. Especially generative AI, a new piece of tech that promises to do all the creative work for us while we sit on the beach. Hurrah!

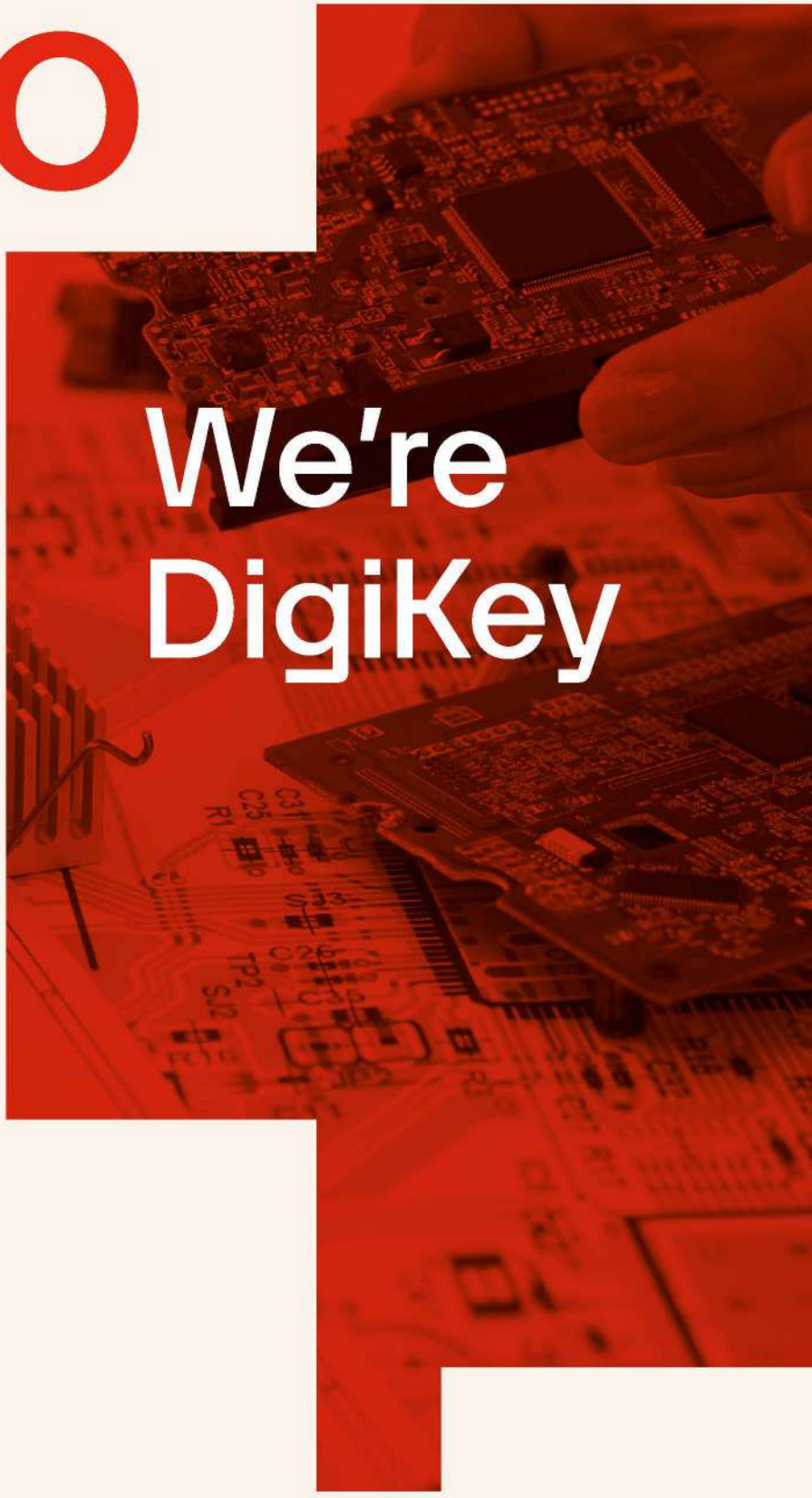
Of course, it's never that simple. There are ethical conversations surrounding generative AI, something that – as a technology – is both complicated and opaque.

Who knows what is going on inside? KG does, that's who. Our Raspberry Pi AI Made Clear feature (**page 30**) demystifies artificial intelligence by showing you how to develop generative technologies using Raspberry Pi and open-source software. Create personal image diffusers, generate large language models, and assemble intelligent-acting robots, cameras, and speech assistants. All while keeping one eye on the ethics at play!

With *The MagPi* magazine you will understand AI without resorting to ChatGPT to hallucinate an answer. You'll learn how AI works! No small skill to have for the years ahead. Enjoy this issue!

Lucy Hattersley Editor

Hello



Whether you're an

- **engineer**
- **designer**
- **purchaser**
- **maker**

we've got the products, services, and business solutions to help move you forward.

Explore and connect today.
digikey.co.uk

DigiKey

we get technical

Contents

Issue 141 > May 2024

Cover Feature

Regulars

Project Showcases

Bechele 3.0

Mt32-pi Atari ST

The MagPi is published monthly by Raspberry Pi Ltd, 194 Cambridge Science Park, Milton Road, Cambridge, England, CB4 0AB. Publishers Service Associates, 2406 Reach Road, Williamsport, PA, 17701, is the mailing agent for copies distributed in the US and Canada. Application to mail at Periodicals prices is pending at Williamsport, PA. POSTMASTER: Send address changes to The MagPi, c/o Publishers Service Associates, 2406 Reach Road, Williamsport, PA, 17701.

Tutorials

Learn Python functions

Control a robot arm

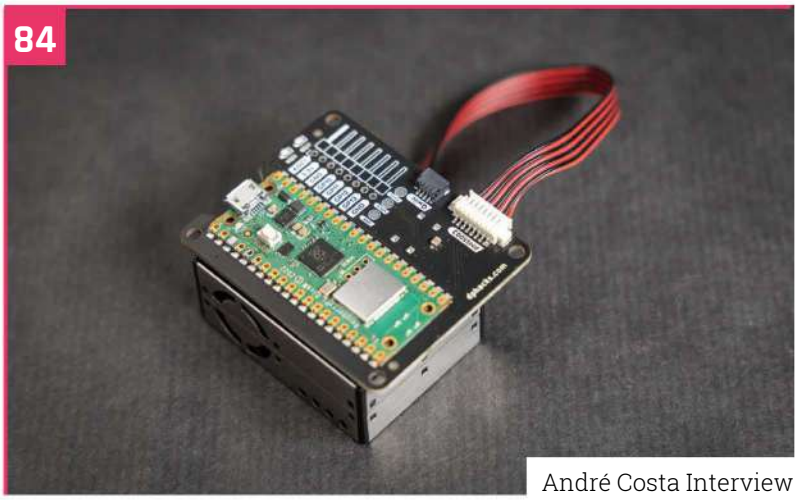
The Big Feature

Raspberry Pi 5 cases group test

Home Assistant Yellow

Reviews

Community

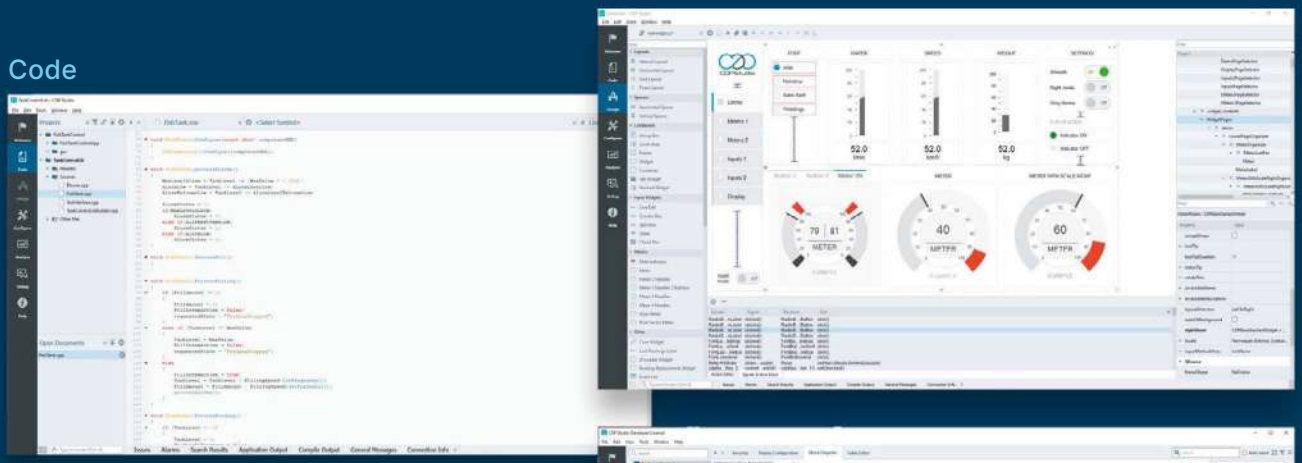


André Costa Interview

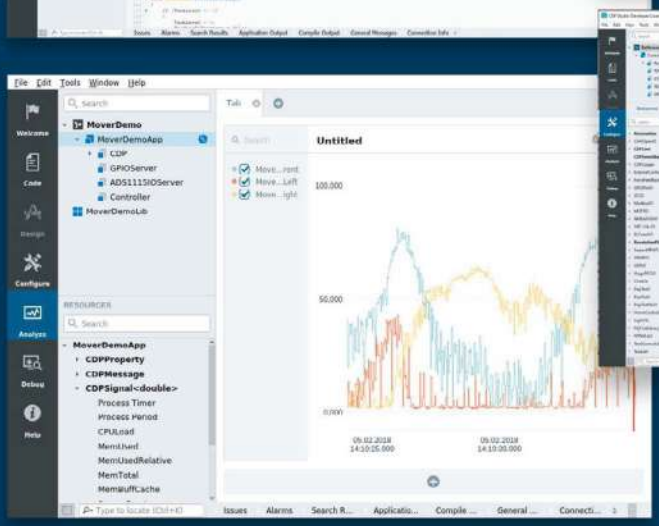
WIN ED-HMI3020 **SCREEN** 94

Design

Code



Configure



Analyze

PROFESSIONAL CONTROL SYSTEM DEVELOPMENT TOOL

Home projects made easy.

CDP Studio, a great software development tool for your home projects. Build systems for Raspberry Pi, use C++ or NoCode programming, open source libraries, out of the box support for GPIO, I2C, MQTT, OPC UA and more. Create beautiful user interfaces. Built for industrial control system development, **FREE for home projects**.

cdpstudio.com
Tel: +47 990 80 900 • info@cdptech.com
CDP Technologies AS // Hundsværgata 8, 6008 Ålesund, Norway

CDPStudio

PiDP-10

After tackling the PDP-8 and PDP11, Oscar Vermeulen goes mainframe with DEC's PDP-10. **PJ Evans** fires up the big iron

MAKER

Oscar Vermeulen

Oscar, 56, lives in Switzerland and likes to spend his time dabbling as a Maker. He has been collecting vintage computers for the last 30 years, going back further and further in computer history.

obsolescence.dev

Over the years, maker and retro enthusiast Oscar Vermeulen has wowed us with his faithful scale models of Digital Equipment Corporation (DEC) hardware. First came the PiDP-8, which paired a Raspberry Pi-based emulator with a meticulously recreated front panel to recreate the 'blinkenlights' experience. A few years later a more ambitious PiDP-11 project introduced a custom injection-moulded case for the PDP-8's big brother. Now, Oscar has completed his most ambitious project yet, a two-thirds replica of the PDP-10.

Before we get to the project itself, we wondered why the PDP range had been recreated in this order. In other words, why make the PDP-10 now? "The first PiDP was a replica of the PDP-8. From 1968, at the time it was the absolute minimum hardware that could be a useful computer. DEC, the manufacturer, actually saw it as we'd now see an Arduino," explains Oscar. "Then, I did the PDP-11 next. It's the forefather of 'everything' we have today. Unix grew up on the PDP-11, so that gets you Linux, Android and macOS today." So while the PDP-8 and PDP-11 shared a 'smaller' computing architecture, the PDP-10 was a different beast altogether, a mainframe.

Mainframes were actually quite boring machines to work with. You created punch-cards, fed them in, waited, then a response was printed out. DEC's PDP-10 was the first to be truly interactive. It was DEC's decision in 1968 to provide a unit to MIT that really sealed its place in history. For the first time, a mainframe became a multi-use computer. Students and staff attached graphics terminals and other displays and started hacking the system. One of the main outcomes from this period was the first computer game, *Spacewar!* PDP-10s were the first computers to connect via ARPANET, which later became the internet.

Pieces of the puzzle

Oscar's new project had two distinct parts: The emulator and the hardware. In truth, the hardware is a simple lightboard with some control circuitry. It's the attention to detail that makes it special. However the software was a bigger problem. Oscar wanted to recreate the MIT experience, and that meant emulating their custom operating system, the Incompatible Timesharing System (ITS). Fortunately, a group was already working hard

The first ever computer games ran on the PDP-10, and you can play them too

Early computers allowed an operator to view their workings, which became known as 'blinkenlights'

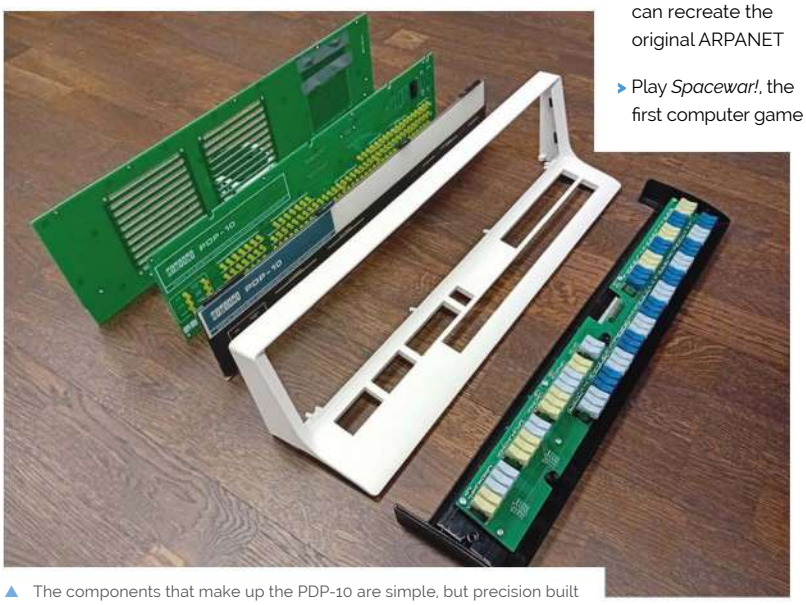
The operator can use a terminal, teleprinter or the key panel to control the computer

Quick FACTS

- Oscar is now ramping up production of kits
- Can also be used as a regular Raspberry Pi too
- Allows you to experience a crucial part of early computer history
- Two or more can recreate the original ARPANET
- Play *Spacewar!*, the first computer game

on emulating ITS, despite MIT having switched everything off in the 1990s. When Oscar met Lars Brinkhoff, the principal driver of the project, the PiDP-10 was born.

Now emulation was taken care of, attention turned to the hardware. Part of the magic of the PDP-10 is its beautiful panel, complete with rows of lights and toggle switches. Oscar was determined to create an accurate replica and found it a satisfying and fun process: "Producing that pretty front panel, with its artwork and transparent windows for the LEDs behind it. That is artisanal work that I really enjoy. Endless tinkering, and then tweaking a big printer to print out the front panels with the exactly correct colours, that sort of thing."



▲ The components that make up the PDP-10 are simple, but precision built

The injected-moulded case, over 50 cm in width, would be more challenging though. "The problem with injection mould making is – you make a 3D model, you give it to a mould maker, and if you're lucky he makes a mould to your specifications. Any mistake you made in your 3D model is your problem, there is no going back to fix things." Luckily, Oscar found a friendly manufacturer who understood the importance of the project and helped him with the process. The result took two years to get right.

The PCB was designed in Kicad, connecting matrices of LEDs and switches to the GPIO of a Raspberry Pi 5, which would be hidden away inside the case. The next challenge was to interface everything together. "Part of the fun is getting the simulator working right. Making sure that it blinks the right LEDs at the right time, that the complex front panel switches for the debugger built into the system really work as they did on the old machine." explained Oscar. But how to confirm this? Since the mid-eighties there have been no operational PDP-10s. In 2018 Paul Allen, co-founder of Microsoft, who had a strong affinity with the mainframe, tasked the Living Computer Museum in Seattle with restoring an example. Shortly before the pandemic, Oscar was able to see the machine working and fine-tune the emulation to match. Sadly Allen died a few weeks before the restoration was complete.

It was obvious to Oscar that the recreated PDP-10 would be based on Raspberry Pi. A primary factor was cost, as he could see no way he could assemble all the I/O ports and connectors necessary for less than the cost of Raspberry Pi 5, which had everything he needed. Even though the Raspberry Pi does not run a true real-time operating system, it is fast enough to be able to scan the switches and refresh the LEDs two hundred times a second, which is more than enough for accurate emulation.

> "Part of the fun is getting the simulator working right. Making sure that it blinks the right LEDs at the right time"

▶ A look at the back reveals the mount for Raspberry Pi 5

Two hearts

Oscar also wanted the result to be 'dual-hearted'. Emulating a PDP-10 and controlling the front panel is a breeze for Raspberry Pi 5, so there's plenty of processing power available to run other things such as a media server. It's certainly more aesthetically pleasing than a small black box.

Is Oscar happy with the result? "Yes! In the end, I think the primary motivator for such a project is that you want to have this thing for yourself, blinking away in your living room, and it becomes almost an obsession to Get It Right. And I think we did." He acknowledges that this has been a project with a lot of contributors such as the ITS Reconstruction Project and many ex-Digital employees, many of which attended the project's launch at MIT in April.

Best of all, if you're happy with through-hole soldering, you can have your own PiDP-10! Oscar has put the kit into production and you can order it from his site (**magpi.cc/pidp10**) and build your very own 1960s mainframe. This project has taken seven years, but Oscar is not done yet. Next on the list is the PDP-1 from the 1950s which would complete the 'evolutionary tree' of interactive computers. We can't wait.

▼ The beautiful injection-moulded case is two-thirds scale and took two years to get right

Build your own PDP-10

01 The PCB is a simple design, but there are a lot of components. Getting LEDs to line up perfectly is challenging, so Oscar has included an extra board that holds them all when soldering.

02 Once the lights and the switches are soldered, it's time to connect them up with a small ribbon cable. This is a good time to test everything before mounting the PCBs.

03 In this step we see the keys mounted into the injection-moulded case. Now the light board will be mounted and Raspberry Pi 5 can be installed using a further backplate.

Bechele 3.0

Keen to delight family members, maker Rolf Jethon used Raspberry Pi to animate a puppet. **Rosie Hattersley** listens in

MAKER

Rolf Jethon

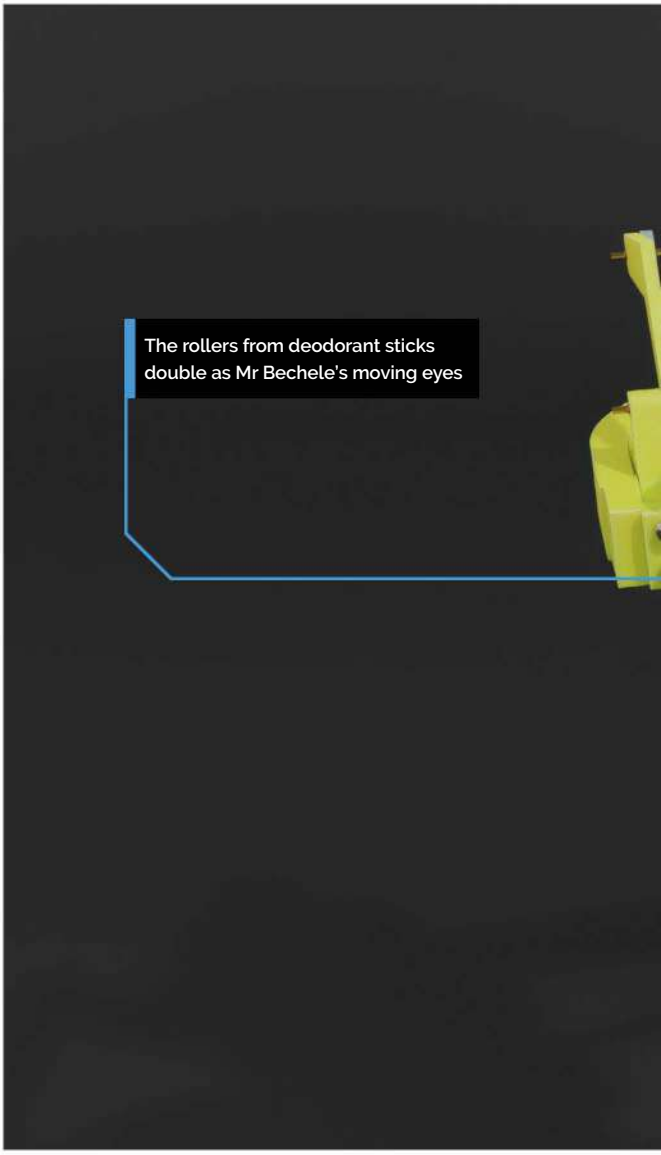
Rolf Jethon enjoys using hardware and software to overcome problems as well as perfecting his clever ventriloquist designs

bechele.de

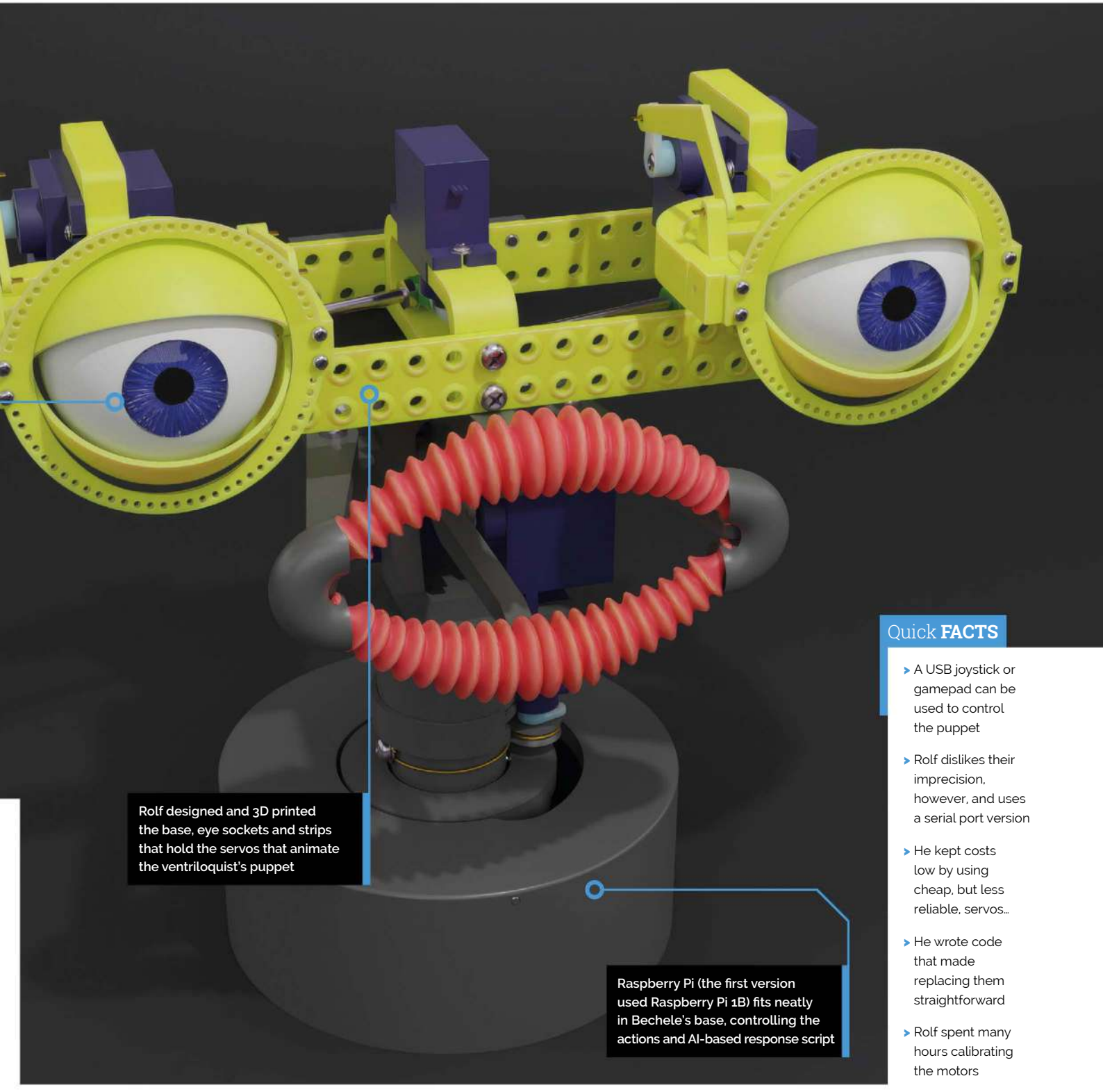
Puppets and play are always popular at children's parties, and the concept struck maker Rolf Jethon as an equally fun diversion for his father's birthday celebration. The hand puppet he made for the occasion was a hit, but Rolf found himself tongue-tied, somewhat muting his own experience: "Though I thought I was prepared well, I forgot part of the text, couldn't master the ventriloquism and the performance stuck several times." As a technical engineer of many years, Rolf decided to use Raspberry Pi to add motion and speech to the existing doll, named Mr Bechele, and have it make another appearance, this time at his mother's 80th birthday party. Although he still had performance issues, things went better once Rolf mechanised some of the actions. Over the subsequent years, he has refined both Bechele's 3D-printed design and the software used to animate him and has just launched Bechele 3.0.

A well-rehearsed speech

A precision mechanics degree followed by a technical career meant Rolf soon encountered and embraced Unix, and latterly Raspberry Pi, both of which came in useful in his free time in which he enjoyed both hardware and software design challenges. "I felt familiar with Raspberry Pi from the beginning: it came with all the well-known tools of Linux, but also provided a cheap, space- and resource- demand reduced way of realising projects."



The rollers from deodorant sticks double as Mr Bechele's moving eyes



Quick **FACTS**

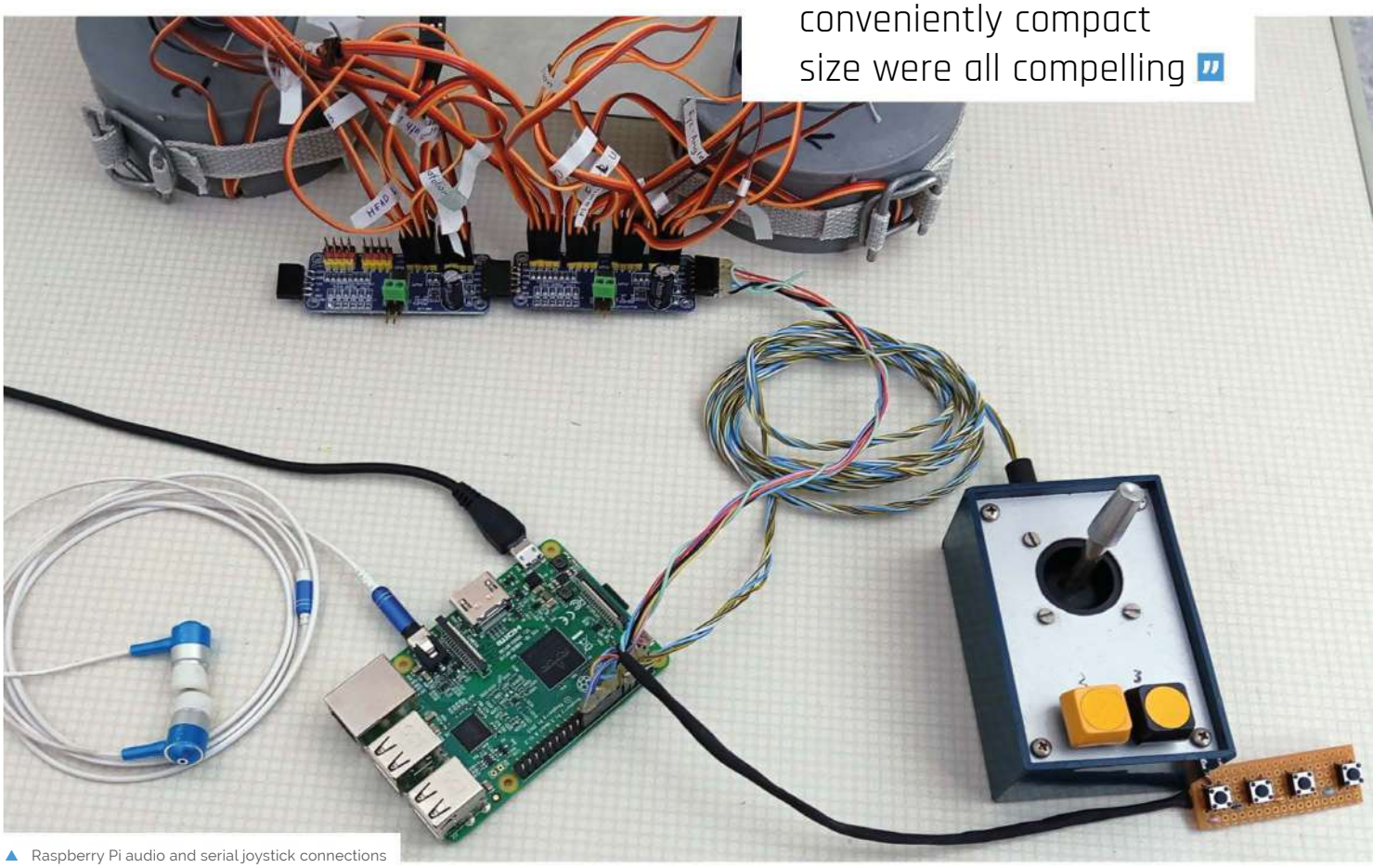
- A USB joystick or gamepad can be used to control the puppet
- Rolf dislikes their imprecision, however, and uses a serial port version
- He kept costs low by using cheap, but less reliable, servos…
- He wrote code that made replacing them straightforward
- Rolf spent many hours calibrating the motors

◀ Rolf's Bechele puppets and their 3D-printed ventriloquist heads

▼ Animating this 24 servo model took roughly two hours

Rolf particularly likes using Raspberry Pi headless and getting it to run things as a 'sophisticated worker' in the background. Using a Raspberry Pi-based setup to turn his hand puppet into a ventriloquist's dummy seemed a natural extension of this idea. The audio outputs, GPIO connections and the conveniently compact size that meant Raspberry Pi could fit inside the puppet's head were all compelling. "The idea was to build a puppet that talks to me and does the face movement on its own. The whole thing should work mainly fully automated [with the] eye, mouth and face movements recorded before, and following a prepared conversation. My task was finally just to ask the puppet the right questions."

> "The audio outputs, GPIO connections and the conveniently compact size were all compelling"



▲ Raspberry Pi audio and serial joystick connections

Resourceful approach

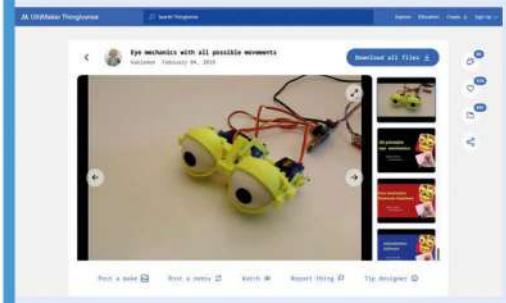
Rolf designed and made any hardware for the Bechele puppets that he couldn't readily source. The eyes are balls from roll-on deodorant sticks with eyelids and sockets, and the control mechanisms are designed in Rhino CAD and then 3D printed. These can be downloaded from Thingiverse (**magpi.cc/eyething**).

However, it was servos that gave Rolf the biggest headache: Bechele uses 24 SG90 servos, so there were challenges to controlling them all, which Rolf partially solved by using PCA9685 PWM circuit boards he'd used successfully in Arduino projects. Resource overheads meant it was very slow driving these using the existing Perl script and the Raspberry Pi i2C connection, but Rolf overcame this by writing his own Perl driver. His code meant the Raspberry Pi could theoretically control up to 64 servos at once.

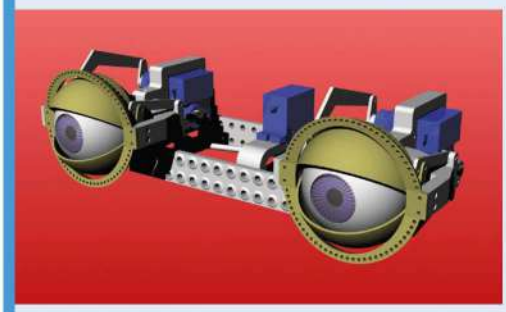
Getting the most from his chosen hardware involved careful and time-consuming calibration of the joystick and recording the movement parameters for each of the servos, but having optimised them, Rolf was then able to swap out any servos as needed easily. He explains that the Bechele software "is fully written in Perl and can be used headless. It loads quickly and has a graphical interface for the teaching process". The number of devices and interfaces involved make installation a challenge, Rolf warns, but he's created a SD card image that anyone can download from **magpi.cc/bechele**.

▼ The heads are fitted with servos and other parts ready for assembly

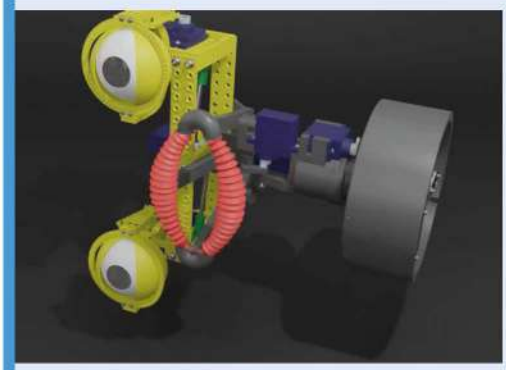
Feeling animated



01 The Bechele blog (**magpi.cc/becheleblog**) explains how to 3D print, assemble and install code to animate a doll or puppet, as well as how to learn the process yourself.



02 Seven servos are used to animate the eyes. Rolf suggests this version (**magpi.cc/eyemechanics**) is simpler to replicate than the one in his main setup demonstration video. The background audio files are used under a Creative Commons licence.



03 Raspberry Pi fits in the base and connects to the 16- or 24-servo configuration via GPIO. Rolf provides downloadable Bechele 3.0 software for models up to Raspberry Pi 4B.

Retro-Printer

Got an old computer or piece of equipment but want to print out graphics and text? **David Crookes** looks at a device that connects retro machines to modern printers

MAKER

Rich Mellor

Rich Mellor, a former solicitor from Walsall, is completely self-taught in computer programming having learned on the Sinclair ZX81 and QL.

retroprinter.com

Many people hate printers and that's putting it mildly. With every paper jam, every automatic update and every message that a single ink colour has run out forcing you to replace the lot, comes much gnashing of teeth and an uncontrollable urge to throw the contraption out of the window. It's why there are rage rooms inside which you can smash the things up with sledgehammers. And yet printers are still useful and that makes them even more annoying.

Acknowledging the printer's worth – grudgingly, we'd suspect – is Rich Mellor, who has had hands-on experience with so many of them over the years, from the humble clackity-clack dot matrix varieties to the slick inkjets and professional lasers. To his frustration, he's noted two things in that time: the expense of buying modern equivalents of old-tech printers, and the lack of compatibility between newer printers and retro computers.

"I was working at Heathrow Airport developing controls software for a baggage handling line in 1999 when we decided it would be useful to attach a dot matrix printer to keep track of errors and how the code was treating each piece of luggage passing through the system," he explains.

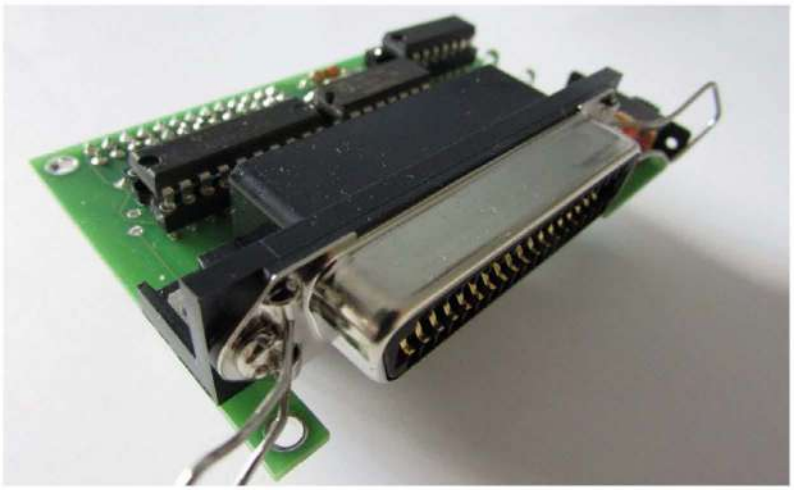
"The choices for a dot matrix printer with a Centronics connector were very limited and still cost more than £500. From using a Sinclair QL at home I also knew the problems of finding a suitable printer which understood plain text and realised this was going to pose an increasing issue for home-computer users and industry."

The Retro-Printer understands both text and graphics created on a wide range of retro computers including this one – an Amstrad CPC 6128 running Protext

Having talked with two fellow electrical engineers, he tried in vain to work out a low-cost solution. "It was only with the release of the earliest Raspberry Pi that I realised an affordable printer emulator could be developed," he says. This prompted him to begin work on the Retro-Printer project – a HAT designed to plug into a Raspberry Pi computer that allows data captured from the

A Centronics cable is needed to connect a retro computer to the Retro-Printer. Rich found it difficult to get a big supply of female Centronics connectors

The Retro-Printer converts data so it can be stored electronically or sent to a USB or network printer (you can connect wirelessly or via USB)



The Retro-Printer HAT means you don't need to buy an expensive dot-matrix printer, delve into the used market (and struggle to find ink cartridges or ribbons) or upgrade your software and equipment

Quick **FACTS**

- It connects to the Centronics port of retro computers
- It allows data to be sent to modern USB and network printers
- A large number of Centronics printers are supported
- The idea originated 25 years ago!
- Around 500 Retro-Printers have been sold

▲ The Retro-Printer software formed Rich's second ever C program. His first was amending third-party code for the Sinclair QL to support 720dpi and full colour printing on an Epson printer in 1999

▶ Emulating the Seiko QT-2100P printer was tricky. It works with a watch-timer instrument, but the control codes didn't follow any logical pattern for text or graphics output. Rich discovered it sends the raw data of tick-tock readings from a watch and the printer decides how to display the readings

Centronics (or parallel) port on a retro machine to be sent to a modern printer.

Bit by bit

Most personal computers released from the 1970s to the 2000s had Centronics ports for connecting the machines to printers and other devices. Data would be sent eight bits at a time in parallel to each other at 150 kilobytes/sec. Rich figured he could develop working hardware that acted as an interface between the port of any one of those computers and the Raspberry Pi. Software would then convert the captured data into a PDF, ready for printing.

"The Retro-Printer HAT is simply an interface to convert the incoming data signals from the Centronics output to the GPIO pins on a Raspberry Pi computer," Rich says. "The hardware incorporates some hard-wired signals so that the busy signal is asserted as soon as any data is received from the Centronics port, and we found that we needed to add jumpers to allow 5v to be enabled on three different pins of the Centronics port as this was expected by some computers. We also initially added an offline switch as a nod to the fact that many old printers had one but we've since changed the software to support different functions on that switch such as swapping page sizes or printer types."

The software is written in C and it makes use of a single capture process that runs constantly. "It watches for incoming data, spooling it to a temporary file and sending the acknowledge and ready signals as we process each byte," Rich continues. "That process also has to deal with 7-bit or 8-bit data, the position of an offline switch and different timing mechanisms."

There are two background processes: "One handles the three LEDs on the HAT which can be controlled externally by the user if they wish and the other is the all-important printer emulator." Since there have been so many different printer

types over the years, testing has been a major challenge. "We've had to work alongside users to identify and address issues particularly around equipment which does not fully implement the Centronics protocol or where there is little or no information about programming."

Ink-spiring

Such is the lack of information about some original printers, Rich has sometimes had to rely on old faded printouts or photos on the internet. "We've been able to reverse engineer the printouts mainly using a Sinclair QL emulator in the first instance to understand what the data means," he says. As a result of all of this, different printer emulators have been developed for each printer type.

"They all include standard classes for reading the captured bytes from the temporary file and graphics routines and creating the PDFs," Rich explains. "The printer emulator is responsible for interrupting each captured byte and determining whether it is data for processing, text or a printer command – for example, an ESCape code sequence.

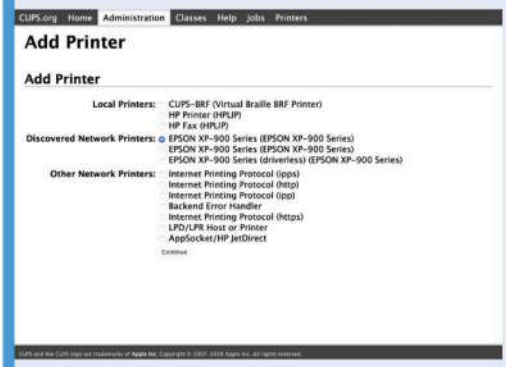
"We then create a PNG of the printed page in memory using those printer commands to set the position on the page and draw any text or graphics. Once the capture process indicates that it thinks the end of the print job has been reached, or we have filled a page of data, the PNG is converted to a PDF for printing or storage."

Given the amount of work involved, it's surprising to hear that the project was created rather quickly. "Thanks to a contact in Germany who had previously developed hardware for the Sinclair ZX81, the initial hardware and basic Epson printer emulator only took six months to develop," Rich says. "Getting the product readyfor manufacture by a well-known Raspberry Pi supplier took a further 12 months, although thanks to a production issue picked up after the initial batch of boards were received and sold, it took another six months before we had a v4 Retro-Printer."

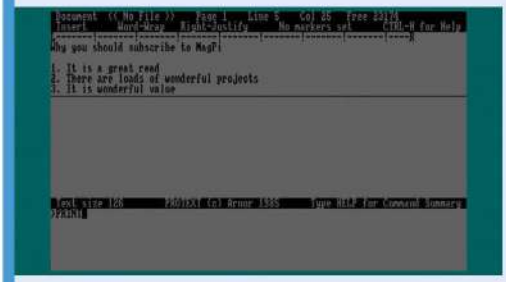
Not that the work is complete. Development is still ongoing as Rich meets new challenges and old printer technologies. "We have found a niche among retro computer users and industry alike, with uses from the humble ZX Spectrum and Commodore 64 through to various commercial flight simulator and industrial production lines," he says. "In some cases, we have extended the useful life of multi-million pound equipment which would have needed complete replacement for the sake of a working printer."

Printing pages

01 Connecting up: fit the Retro-Printer HAT to Raspberry Pi, install the Retro-Printer software, connect an Ethernet cable between your router and the device, find the Retro-Printer's IP address and set up a wireless LAN connection.



02 Add a printer: plug the device into your retro computer. Find the Retro-Printer's IP address and connect to the CUPS Linux printing engine. Once in CUPs, look for and add your printer (loads are supported). Set it as the server default.



03 Send a job: now the fun begins. Use your retro computer to create a document and instruct the machine to print it. The data will be captured by the Retro-Printer as a PDF and it can either be stored on the device or printed out.

Mt32-pi Atari ST

Reviving an old synthesiser using Raspberry Pi was a labour of love for *The MagPi*'s publisher, he tells **Rosie Hattersley**

MAKER

Brian Jepson

Brian is *The MagPi*'s publishing director and a volunteer, and restorer at Rhode Island Computer Museum.

jepSTone.net

Alongside his day job as *The MagPi* and *Hackspace* magazines' publisher, Brian Jepson likes to track down and restore old computers. As a volunteer at Rhode Island's Computer Museum (**ricomputermuseum.org**), he regularly rescues old computers "from 8-bit on up". A favourite is the Atari ST line of 16/32-bit computers which launched in 1985, featured Motorola's 68000 CPU range and, "uniquely, could be used with electronic synthesisers as well as in home computers". Brian's mt32pi synth project shows just how well the 40-year-old hardware works with a thoroughly modern Raspberry Pi.

Musical inspiration

Brian used to enjoy playing music (and regrets not having kept it up) and was keen to add a vintage synthesiser to his collection. He observes that the Atari ST's MIDI ports were notable because certain videogames from the era were capable of producing high-quality background music with a MIDI synthesiser. "There were many music sequencers you could use to compose music with a MIDI keyboard and play it on a MIDI synthesiser.

> Brian's mt32pi synth project shows just how well the 40-year-old hardware works with a thoroughly modern Raspberry Pi

However, the Atari ST was unique in that it had built-in MIDI ports for connecting to electronic musical instruments as well as synthesisers." There are also several MIDI-capable keyboards that were designed as videogame controllers that, for some reason, include the five-pin MIDI DIN cable, Brian continues. "For example, the *Rock Band 3* Wireless Keyboard for Nintendo Wii is a relatively affordable keyboard that works great with this setup! It's unusual to see the five-pin connector because a lot of modern MIDI stuff uses USB now."

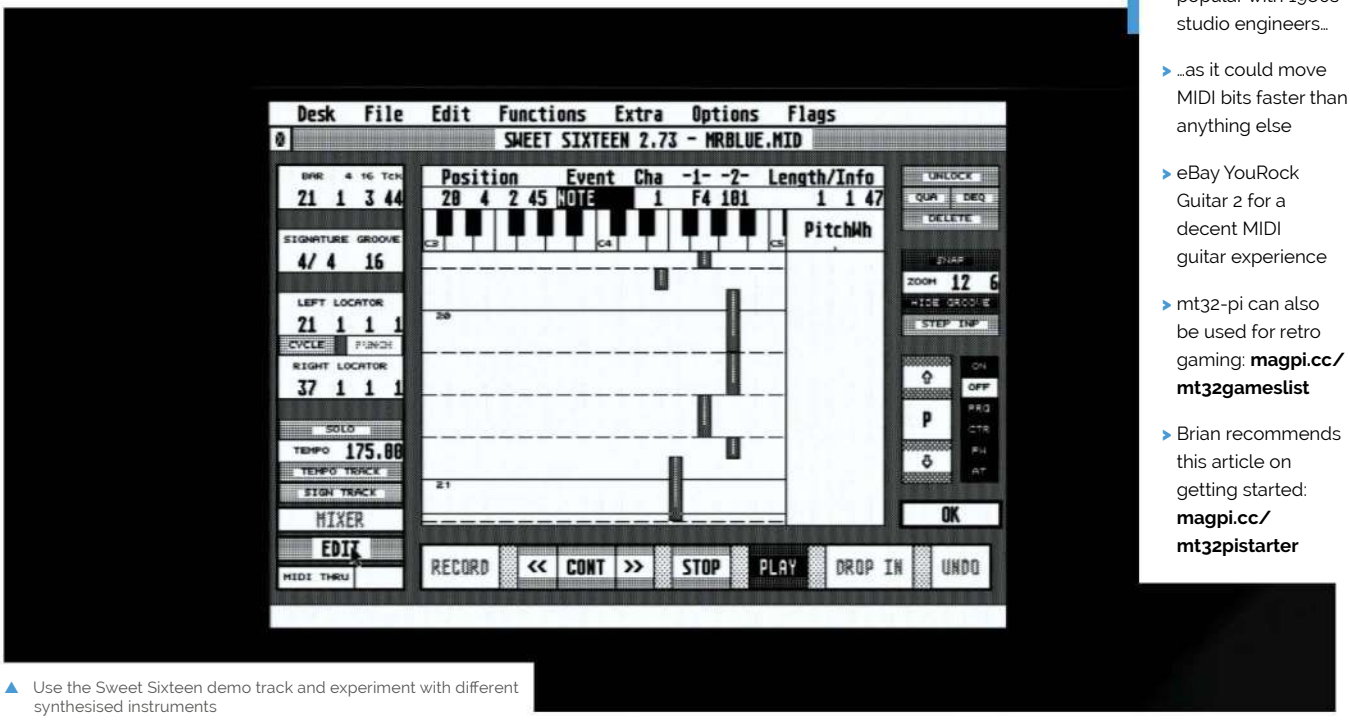
For his own needs, Brian "looked into buying a used Roland MT32 synthesiser, which was roughly contemporary with the Atari ST, but decided the versatility of a Raspberry Pi-based solution was preferable". For his Atari ST revival, Brian used the ClumsyMIDI board with mt32-pi

▲ The mt32-pi board with a five-pin cable

The ClumsyMIDI board provides a MIDI interface, digital audio processor and an OLED display. Several other boards also work with mt32-pi: **magpi.cc/mt32piwiki**

mt32-pi is an open-source synthesiser emulator that works with Raspberry Pi 3, 4 and Zero connected to the Atari's MIDI ports

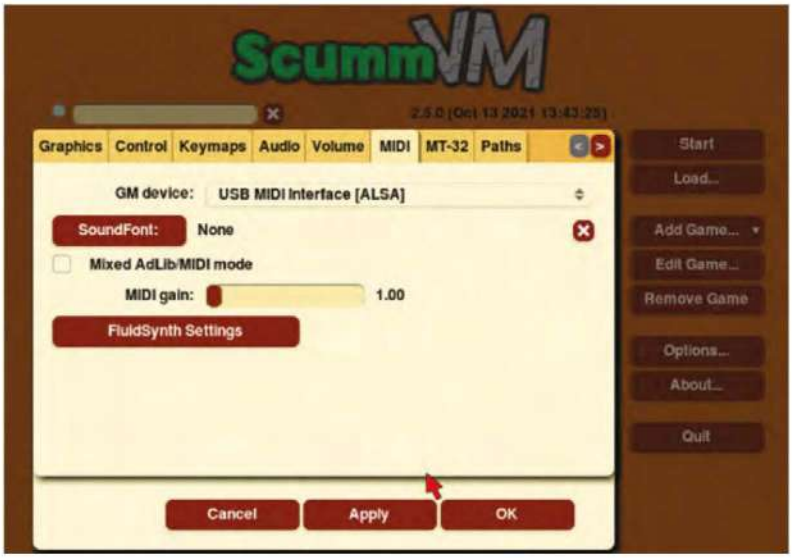
The 1980s Atari ST MIDI synth worked with both keyboard and electronic instruments, making it popular for use in games as well as with touring musicians



▲ Use the Sweet Sixteen demo track and experiment with different synthesised instruments

Quick **FACTS**

- Atari ST was popular with 1980s studio engineers…
- …as it could move MIDI bits faster than anything else
- eBay YouRock Guitar 2 for a decent MIDI guitar experience
- mt32-pi can also be used for retro gaming: **magpi.cc/mt32gameslist**
- Brian recommends this article on getting started: **magpi.cc/mt32pistarter**



▲ Download *The MagPi* 114 for a mt32-pi MIDI setup guide (**magpi.cc/114**)

▶ Brian's PiSCSI emulator came in handy for an Atari project too

Warning!
Hot solder

Soldering irons get very hot, and stay hot for a long time after they're unplugged. Make sure that you put the iron in the stand when you're not using it and don't touch the metal parts – even after it's unplugged.

magpi.cc/soldering

software (**magpi.cc/mt32piwiki**) – an MT-32 emulator that works with Raspberry Pi 3, 4 and Zero. Developer Dale's wiki explains the project in detail: it was ready to go and worked well with Brian's Raspberry Pi 3Bs and "a couple of Ataris I've outfitted with mt32-pi synthesisers".

Fast and affordable

Dale Whinham's open source mt32-pi software (**magpi.cc/mt32pigit**) is a bare-metal MIDI synthesiser that operates as a kernel. It boots directly into the mt32-pi software, starts up in seconds, doesn't mind being unplugged without the usual shutdown process, is extremely fast, and is far more versatile than any alternative Brian could find.

Next, Brian needed to decide which board would work best for the Atari ST rebuild (see **magpi.cc/mt32picustom**). He chose ClumsyMidi (**magpi.cc/clumsyMIDI**) because it is based entirely on through-hole parts which he was able to order along with the Clumsy circuit boards.

Aside from the Atari ST, Brian estimates the project cost roughly $75 and came together very easily. There was one aspect that left him a little dissatisfied though. "If you're planning to mess around with making music on your Atari, you will have the most fun if you can compose using a real musical instrument rather than just using the software. So you'll want to build some practice and learning into your time budget. I say this as someone who has neglected that part and regrets it."

Brian's Atari ST is used for both retro games and MIDI demos at the Rhode Island Computer Museum, where visitors can connect it to a keyboard and try out different instruments playing the freeware Sweet Sixteen sequencer (**magpi.cc/s16atari**). "It's amazing how much fun someone can have switching instruments and banging on keys."

Making MIDI

01 You don't need a vintage synth to try out mt32-pi. It can be used with a USB MIDI interface.

02 If you are making your open hardware setup, you need to solder the ClumsyMIDI (or other chosen) board together, download the mt32-pi software from **magpi.cc/mt32pigit** and install it on to a spare microSD card.

03 Place the MIDI board atop the Raspberry Pi and connect the five-pin DIN MIDI cables. Power on, and mt32-pi should boot within moments. You're ready to start experimenting with electro-synth music.

CoolCoral Project

How do you cool ocean coral? A group of volunteers discuss this regularly and their current solution makes use of Raspberry Pi Pico. **Rob Zwetsloot** takes a look

MAKER

Iestyn Jones

Iestyn is an engineer with an interest in climate science. He's part of a volunteer group of people seeking to help researchers maintain coral for study.

magpi.cc/coolcoral

As the climate around the globe changes, nature itself is being affected. Ocean reefs and their coral are just one of the ecosystems under threat, so research into them is very important.

"Field experiments involving coral in tropical climates are increasingly hampered by heatwaves – and that's a pleasant turn of phrase," Explains Iestyn Jones, a member of the volunteer group CoolCoral Project. "Basically, unpredictable heatwaves cause unpredictable temperature shifts that can overwhelm and, more often than not, kill coral samples and render oceanographic experiments fruitless."

These fatal heatwaves are becoming more regular due to global warming, however the device the group have been working on "concentrates cool water onto coral samples and alleviates the fatal effects". Water cooling for coral then, like a high-spec gaming PC.

"It'll be a solar-powered machine that'll be cooling sea-water temperatures and running a Raspberry Pi Pico for data readings and collection," Iestyn tells us. "It's simple science –KS4, I'd estimate – but it's a combination of ideas that's gone untested… until now!"

Pico at sea

Members of the team described the build process as "agile, rapid prototyping," so it's no wonder they chose Pico for the job.

"Not only is Raspberry Pi Pico extremely accessible and uncomplicated, but it offers a comprehensive range of functions and capabilities that our device requires," Iestyn says. "For example, Pico's ability to run two different threads simultaneously allows us to run the device itself, its pumps and whatnot, while also hosting a website."

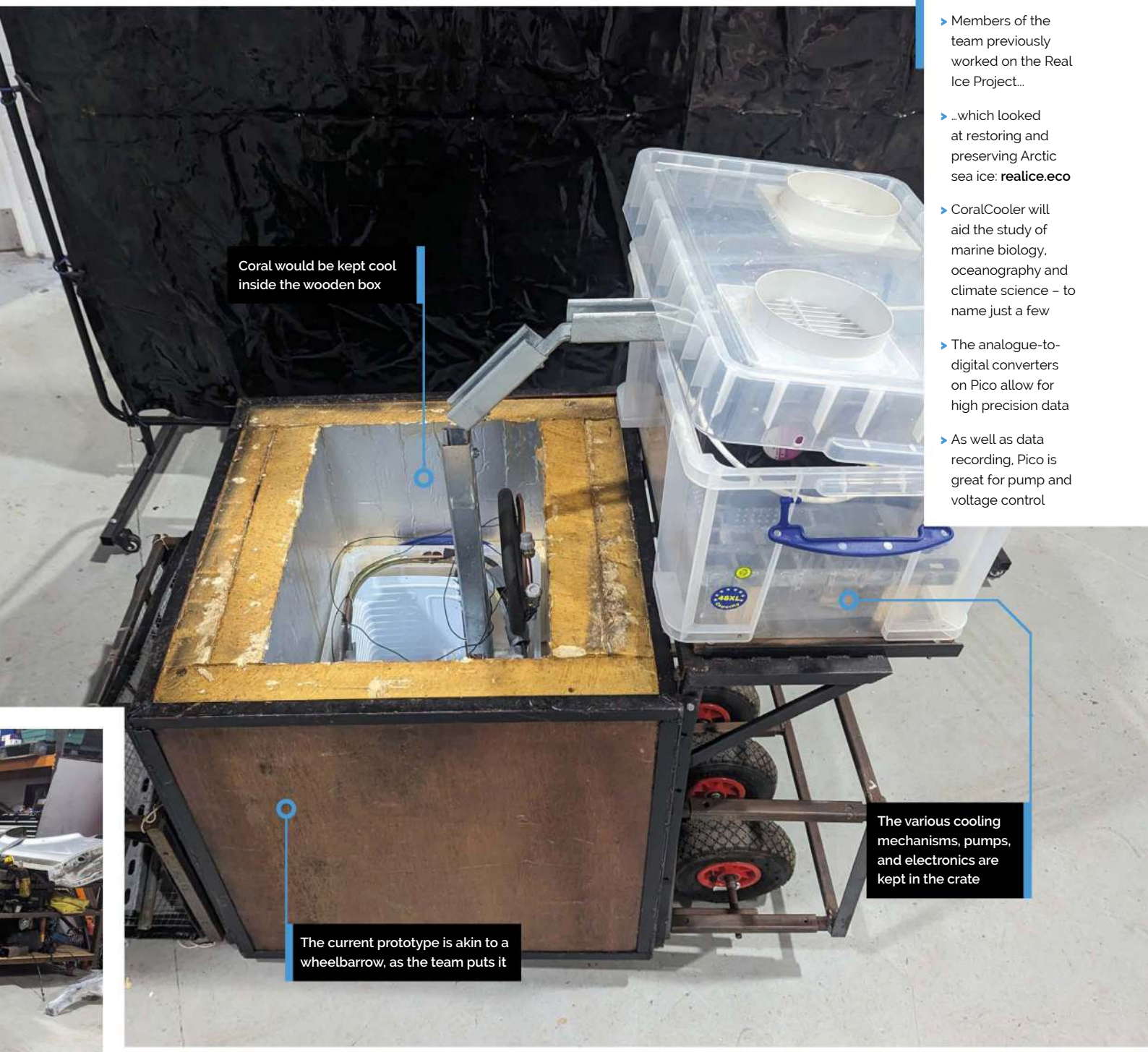
The web server is accessible remotely, not only allowing easy access to data across the globe, but also to update the system thanks to the wireless connection on Pico W. The system itself does a lot more than take readings, though.

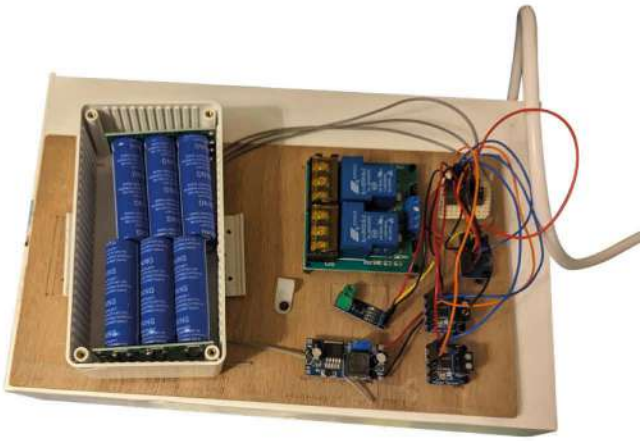
"From the outside, the device looks like a wheelbarrow drawn by a three year old: it's a cumbersome brown box with handles and some dinky wheels but, inside, it's a toybox of entropy," Iestyn says. "Pumps take in seawater through a hose before sending it through a digestive tract of more hose, copper pipes and a ribbed-metal refrigeration unit that's filled with ice. It doesn't need saying that the water cools on this journey.

▶ The solar panels are currently not equipped, but would provide power to the system

Quick FACTS

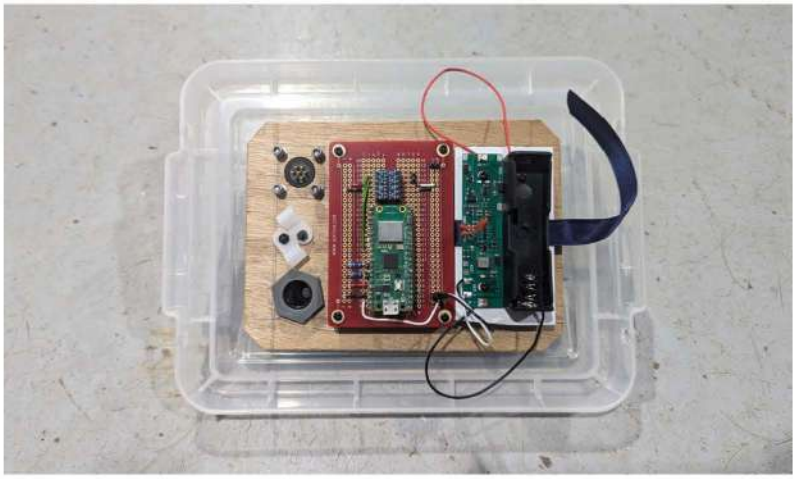
- Members of the team previously worked on the Real Ice Project...
- ...which looked at restoring and preserving Arctic sea ice: **realice.eco**
- CoralCooler will aid the study of marine biology, oceanography and climate science – to name just a few
- The analogue-to-digital converters on Pico allow for high precision data
- As well as data recording, Pico is great for pump and voltage control





▶ Serious battery power is required to keep the system going without solar panels

▼ The development involved a lot of rapid prototyping, a perfect use for Pico



> The device the group has been working on concentrates cool water onto coral samples and alleviates the fatal effects. Water cooling for coral then, like a high-spec gaming PC

Eventually, the cooled seawater enters another external pipe that runs towards the seabed and, hopefully, a chunk of coral that's shivering its timbers over the prospect of a heatwave.

"Meanwhile, floating above the sea level in a hermetic box is our Raspberry Pi Pico. Up there, [Pico] is running constant temperature and performance readings which are being sent in real time towards a device connected to the Wi-Fi. If anything goes awry, a notification is zapped across the air and the device can be reeled in for repairs."

The team has made sure that the mechanical aspects of the device can be fixed as easily as possible, in fact all it requires is a single type of spanner. "The whole thing, meanwhile, is being powered by two enormous solar panel wings held afloat on either side of the device by rubber tyres, or buoyancy aides." Iestyn says.

Prototyping stage

The device is still undergoing tests, with most of the electrical and mechanical system done. Currently float tests are being conducted.

"If anything [it was] too successful," admits Iestyn. "With the device floating much higher than it needed to and answering the question: 'Will we require ballast?'"

▶ Testing is meticulous to make sure everything works perfectly

Once all the testing and prototyping is finished and combined into a single test, the team reckon they'll start on a second model in the future based around their findings.

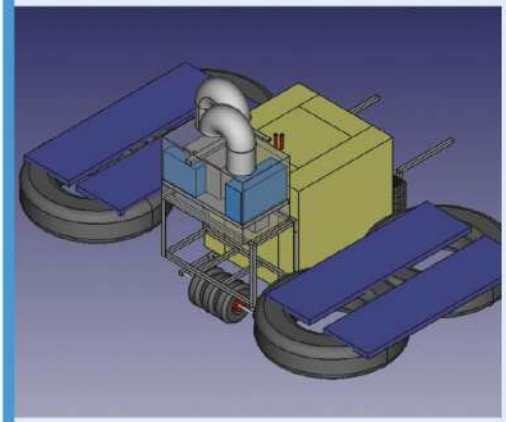
"We're conscious that the CoolCoral Project is coming across as oppressively niche because, well, it's a machine that chills-out diddy clumps of coral – it's hardly carbon capture technology," Iestyn says [*Ed note: we think it's very cool*]. "However, the project is terribly exciting because of its flexibly multifaceted nature. There's already interest being shown towards utilising this device for experimenting with substrate microbes, and that interest isn't even concerning the coral-cooling part of it. A floating device that has the potential to store masses of energy is, frankly, unheard of in ocean sciences. Most ocean-based experiments running today require boats and diesel engines to run. Our machine uses solar panels! This is how untouched this kind of thing is! We're tinkering on the tip of a vast iceberg of possibilities when it comes to ocean-centric energy devices, and we're very excited to see where further experimentation can take us."

▼ Ballast can be added to the metal frames at the front of the device

Cooling coral

01 Get the device into the water and prepare the coral sample you need to cool down.

02 Water is pumped through a refrigeration system and cooled before being pumped to the coral to make sure it too keeps cool.



03 With enough power this cycle can be repeated indefinitely, which is where the large floating solar panels will come in eventually.

SUBSCRIBE TODAY FOR JUST £10

Get 3 issues + FREE Pico W

+

Subscriber benefits

- **FREE Delivery**
  Get it fast and for FREE
- **Exclusive Offers**
  Great gifts, offers, and discounts
- **Great Savings**
  Save up to 23% compared to stores

Subscribe for £10

- Free Pico W
- 3 issues of The MagPi
- £10 (UK only)

Subscribe for 6 Months

- Free Pico W
- 6 issues of The MagPi
- £30 (UK) $43 (USA)
  €43 (EU) £45 (Rest of World)

Subscribe by phone: **01293 312193**

Subscribe online: **magpi.cc/subscribe**

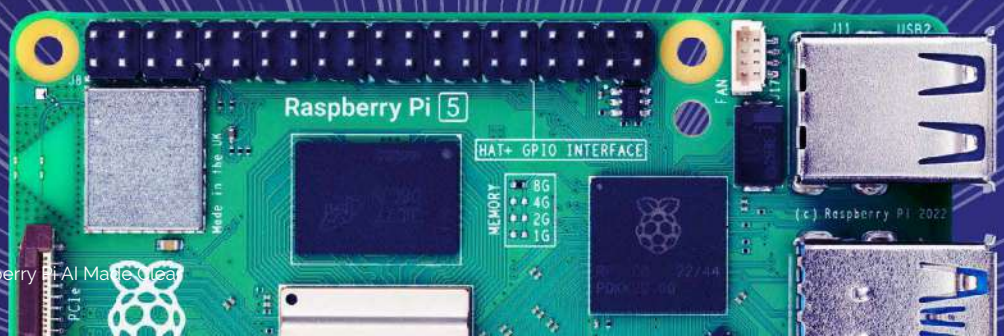
Email: **magpi@subscriptionhelpline.co.uk**

Subscribe for £10 is a UK-only offer. The subscription will renew at £15 every three months unless cancelled. A free Pico W is included with a 6-month subscription in USA, Europe and Rest of World.

RASPBERRY PI AI MADE CLEAR

GENERATE IMAGES WITHOUT INFRINGING COPYRIGHT, ADD NATURAL VOICES TO YOUR SCREEN READER, AND MEET A ROBOT DOG IN OUR GUIDE TO MACHINE LEARNING TOOLS FOR RASPBERRY PI

MAKER
K.G. Orphanides
K.G. is a writer and developer
@owlbear



Machine learning isn't all confabulating chatbots and copyright-infringing image generators, but you'd be forgiven for thinking that in the current AI hype cycle. In this feature, we'll look at your best choices for non-exploitative machine learning projects, fun toys and genuinely useful tools.

We've previously looked at chatbot creation (*Build your own GPT Chatbot*, **magpi.cc/129**), both deploying popular online tools such as OpenAI's very much not open GPT models, as well as fully local, privacy-centric approaches to creating chatbots using models like Stanford University's Alpaca (fine-tuned from Meta's LLaMA).

Some LLMs (**large language models**) include scrapes of the open internet, including web pages containing copyrighted material, GitHub repositories not having their licences respected, and were even found leaking personal data – something the LLaMA 2 training set was scrubbed of.

We'll steer clear of problematic models in this feature and do our best to ensure that we don't use any other models that contain unpleasant surprises. We've instead sought out open and public domain-based projects that can be run locally on your Raspberry Pi hardware.

This approach allows us to understand more about machine learning and AI tools while minimising potential harm.

The ethics of AI

Much has been made of a hypothetical threat that general AI (the popular term for a 'true' artificial intelligence – however you choose to define that) could present to humanity's existence.

This acts as a smokescreen for real ethical issues users in the field of machine learning must grapple with: the high power consumption involved in training AIs and using cloud-based models; unsustainable water use by the data centres where they're trained and used; worker exploitation in both labelling training data and content moderation for live systems, and copyright abuse in the form of model training data scraped from the open internet without consideration of licensing terms.

Users are led to expect authoritative answers, and many appear to assume LLMs in particular will contain current information, even though they can't regurgitate anything that happened after the model's training data cut-off date.

More significantly, what LLMs do is place words in a probable order based on their training – you might have heard them referred to as 'stochastic parrots'. LLMs aren't given labelled training data but are instead fed a vast corpus of writing, and practice emulating the patterns therein.

LLMs create probable text sequences, the result of a training process in which the model attempts to accurately predict the next word in a huge range of texts. What it spits out can be entertaining and even impressive, but an LLM can't analyse what you're asking, or its response. So we see lawyers citing cases that don't exist, chatbots spitting out confabulated stories of criminal acts, recommending made-up software packages that introduce supply chain vulnerabilities, or simply repeating popular misconceptions and plausible but inaccurate answers to unfortunate users who mistake them for a search engine or information repository.

> HOW TO USE MACHINE LEARNING TOOLS WITHOUT BEING GROSSLY UNETHICAL ABOUT IT

Harm reduction

So, how can we use machine learning tools without being grossly unethical about it?

For a start, we'll be running everything locally on Raspberry Pi. That immediately limits your power consumption to about 6W (with Raspberry Pi 5) and ensures that you can keep your data to yourself.

We'll do our best to use machine learning models and projects that use only public domain or consensually collected, opt-in, training data, and highlight any potential issues with others we mention. We'll similarly prioritise open models, where both training data and methodologies are publicly available, rather than opaque black boxes.

And while it's hard to avoid mentioning at least a couple of machine learning tech behemoths, we want to explore projects that are more about small-scale ingenuity than speculative profits and influence for large tech firms.

While 'artificial intelligence' is a misnomer at best, the technologies that get bundled under the name are often fascinating. It's also worth understanding what's behind the latest craze sweeping the industry, and there's no better way to learn than by doing.

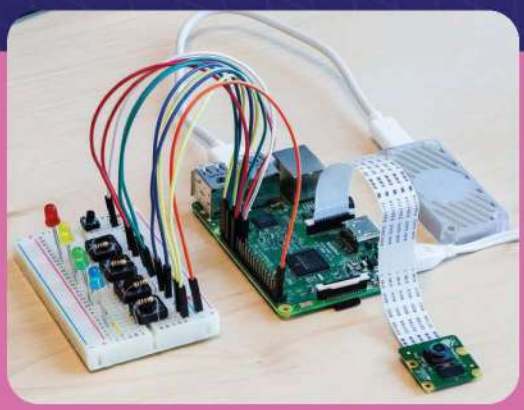
Warning!
Copyright

We've used copyright-safe models for image generation throughout this feature, but most other diffuser models are trained on work scraped without permission and may infringe on living artists' copyright. Read the section Text and data mining for non-commercial research.

magpi.cc/copyright

AI PROJECT SPOTLIGHTS

THESE BUILDS HELP YOU BRING AI INTELLIGENCE TO THE REAL WORLD

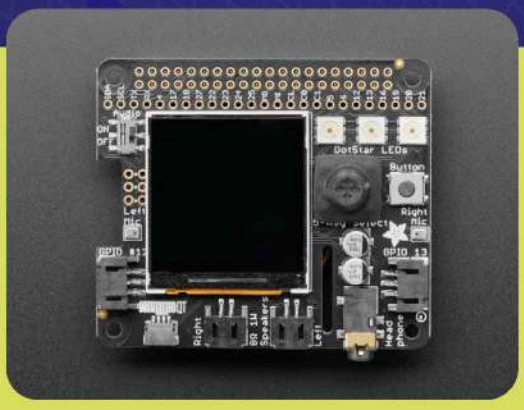


Coral USB Accelerator

magpi.cc/coral
magpi.cc/teachmachine

Coral's USB Accelerator lets you build AI capabilities into Raspberry Pi projects. The accelerator is built around Google's Edge TPU chip, an ASIC that greatly speeds up neural network performance on-device.

We used Coral to build a Teachable Machine, a device that can be taught to recognise objects using Raspberry Pi, Coral USB, and a Raspberry Pi Camera for a complete system that's perfect for executing complex computer vision tasks such as object recognition. Since the Accelerator operates locally, you do not need to connect to a cloud service or share secure data over the internet. It also runs with less latency than a cloud connection, performing object detection in near real time.



Adafruit Braincraft HAT

magpi.cc/braincraft

Built for Raspberry Pi 4, Adafruit's Braincraft HAT is a pretty comprehensive kit for machine learning. It includes a 1.54-inch 240×240 TFT screen as a display, a joystick to navigate options you might wish to display on it, a cooling fan, three controllable LEDs, plus speaker, headphone and microphone ports to help you build text-to-speech tools or home assistant projects. There's no integrated camera, but it does include a slot to connect one, as well as a range of other IC2 and JST STEMMA devices.

Adafruit publishes a range of projects targeting the Braincraft, and is currently updating its guides and software to support Raspberry PI OS Bookworm. The legacy version of Raspberry Pi OS Bullseye, which is known to be stable with the HAT, is available via both Raspberry Pi Imager and as a direct download.

LUWU DYNAMICS' XGO MINI 2 AND LITE 2 DOGS LOOK LIKE DESK-SIZED TAKES ON BOSTON DYNAMICS' SPOT ROBOT

Vizy Camera

magpi.cc/vizycam

Like XGO, Vizi camera is driven by a Raspberry Pi, although it's a standard version, rather than the Compute Module form factor. You can even choose how much RAM you'd like your camera to have.

Once set up on your network, it throws up a local intranet page that you can access from your browser. You can also configure it to be available over the internet. Here, you can access the cam's feed and run a range of built-in machine learning applications including object identification, motion analysis, a bird species identifier for your bird feeder and a pet detector that can take photos or be used as a trigger for other events. Suggested projects include a treat dispenser or ball launcher.

As you'd expect for something that can be used to spy on garden birds, Vizy can be weatherproofed with an outdoor enclosure. Other add-ons include extra lenses, a 4G adaptor for enhanced portability, and a Power over Ethernet adaptor.

Luwu Dynamics XGO 2

magpi.cc/xgo2

Luwu Dynamics' XGO Mini 2 and Lite 2 dogs look like desk-sized takes on Boston Dynamics' Spot robot. The robot can be addressed by a variety of microcontrollers and single-board computers, but the default configuration uses Raspberry Pi CM4.

As well as manual control options, it has a selection of pre-programmed behaviour routines that you can invoke, including object recognition using the Yolo real-time object detection system, voice command recognition, and efforts at gender, emotion and gesture recognition. All of these behaviours are handled by a clutch of Python scripts and their accompanying libraries, which makes them really easy to customise.

If you don't want to immediately dive into XGO's scripts, there are also a range of programming interfaces available and custom libraries for controlling the cyberdog body. Your options range from a remote control smartphone app to a Python-based block-based programming interface accessible via a web browser.

TRANSCRIPTION AND SPEECH SYNTHESIS

One of the most genuinely beneficial developments to have come from machine learning is the ability to run offline tools that can provide sophisticated, accurate text transcriptions of spoken word audio and which can read text for you.

While the latter has unfortunately led to a boom in annoying synthetic-voiced videos, it's genuinely useful when under your control. We're going to install Speech Note (**magpi.cc/speechnote**), a capable neural speech synthesis text-to-speech (TTS), speech-to-text (STT) and machine translation tool that runs entirely locally to ensure your privacy, and set up the Orca screen reader with Piper TTS to give the accessibility tool a more modern feel. First, we'll set up Flatpak and install Speech Note via the terminal.

Speech Note transcription

```
$ sudo apt install flatpak
$ flatpak remote-add --if-not-exists flathub
https://flathub.org/repo/flathub.flatpakrepo
```

Now reboot Raspberry Pi, then open a terminal and type:

```
$ flatpak install speechnote
```

Say "yes" to confirm that you wish to use the suggested ref, that you wish to install it, and that you wish to proceed with the suggested changes to your system installation.

Open SpeechNote from the Sound & Video menu, then go to the Languages menu, search for English (and any other language you may require) and select the models you wish to use.

OpenAI's Whisper is open-source, and transcription tools don't carry the intellectual property violating baggage of text or image generators, so English Whisper Small should do the trick for Speech To Text.

A number of Text To Speech models are available. We find Piper Alba Medium to be both easy on the ears and conveniently near the top of the list.

A few translation options are also available, but this list is limited to only one per language pair. You can try them out based on your translation requirements – we installed English to French and French to English – the latter was only available when we selected French in the first step of the Language installation workflow. Some punctuation checkers are also available, which can be helpful for language learners.

Download the following test file, which is four seconds of speech: **magpi.cc/audiotestwav**.

In Speech Note, click File > Transcribe a file, and open the file we just downloaded. Transcription will begin automatically, and the transcribed text will be output in Speech Note's main text pane.

Integrate Piper with Orca

Piper speech synthesis is a clear upgrade from the eSpeak synthesised voices we're all familiar with. It's not yet available as a default option for many screen readers, but we can integrate it with Orca, the most mature Linux screen reader.

Using Orca with Piper has its limitations, such as comparatively slow reading performance when working in a command terminal and requiring more system resources, but the more natural voice can make it more pleasant to work with if you're primarily interacting with a GUI via keyboard shortcuts.

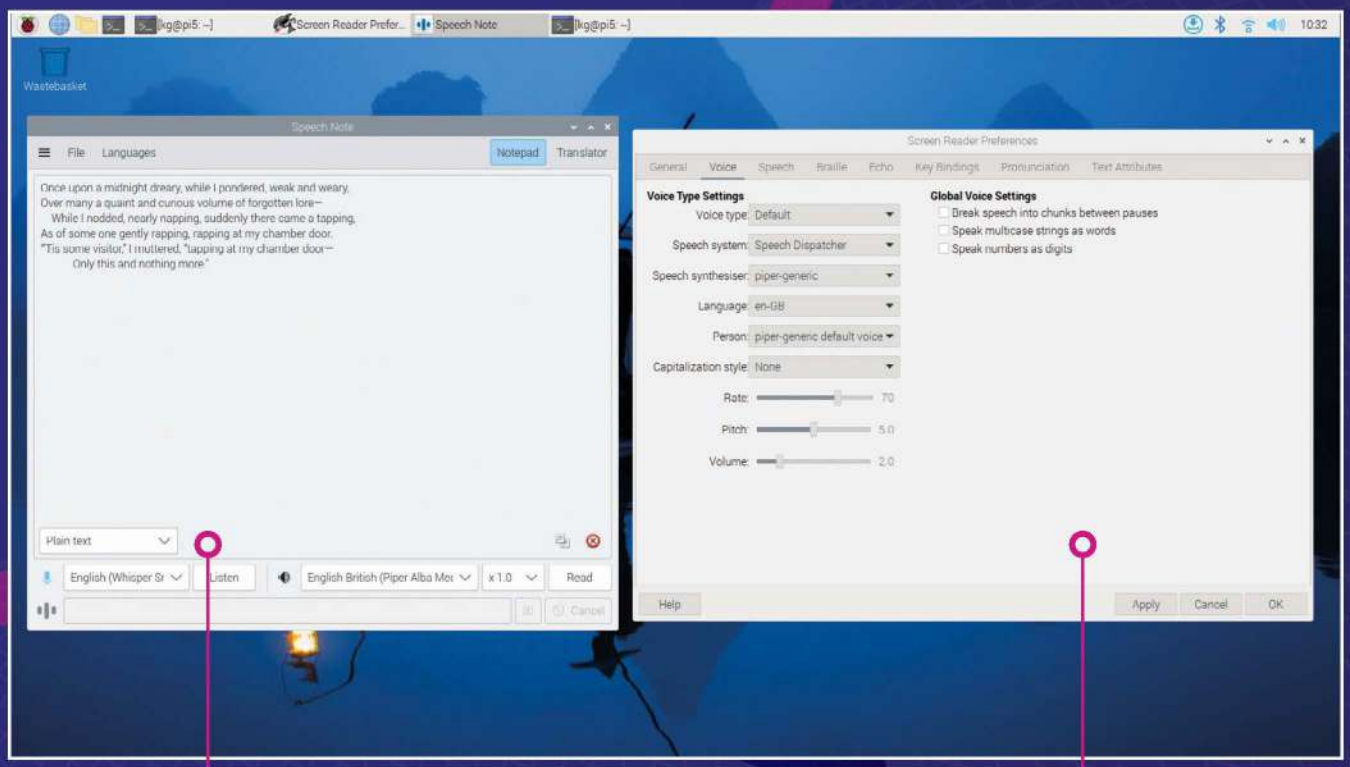
```
$ sudo apt install orca
$ cd Software
$ git clone https://codeberg.org/
MightyOwlbear/rpi-piper-tools.git
$ sh install-spd-piper.sh
```

Enter your password to install dependencies when prompted.

Once installed – assuming the test audio was produced, it's time to configure the Orca screen reader to use Piper:

```
$ orca -s
```

In Orca's settings tool, select the Voice tab. Under Speech synthesiser, select 'piper-generic



Speech Note is a single, fully local application for tools such as Whisper transcription, Piper TTS and translation

We're using the Orca screen reader, to which we've added Piper neural Text-To-Speech

> WE'RE GOING TO INSTALL SPEECH NOTE, A CAPABLE NEURAL SPEECH SYNTHESIS TEXT-TO-SPEECH MACHINE TRANSLATION TOOL

default voice' from the pull-down menu.

Untick the 'Break speech into chunks between pauses' box on the right, then click Apply. You may also wish to change the Rate at which it speaks – we like putting this at 70.

You'll now be hearing a Piper voice named Alan narrating your interactions. It's worth noting that, if you're a quick typist, Piper will not be able to keep up with your letter-by-letter text entry in the same way as eSpeak.

Orca works with both Xorg and Wayland, but we've seen slightly better performance in X. If you're booting to command line and starting the GUI from there, simply invoke

```
$ startx
```

instead of

```
$ wayfair
```

If you're booting to the GUI, you'll have to change your config to reflect your preference. In a terminal type:

```
$ sudo raspi-config
```

Go to 6 Advanced Options > A6 Wayland > X1 X11 and select OK. Reboot to enact your settings changes.

BUILD AN ARTIST-RESPECTING IMAGE DIFFUSER

You'll Need

- 8GB RAM (Note: may not work on 4GB models)
- Large microSD card (64GB minimum)
- Mitsua Diffusion One **magpi.cc/mitsua**
- Python 3.10.6 (Note: Newer versions don't support PyTorch)

Image generation is a minefield if you're hoping to keep your use of machine learning ethical and cost-effective, as well as fun. Services such as Midjourney and Dall-e are black boxes to their users (you can't see what is going on inside). Buying time on these services is costly to you, and also costly to the environment in terms of not only training the models in the first place but also powering and cooling the data centres where they are run.

Many image generators also built on the work of living artists, without their consent, making them a huge problem. Fortunately, we're starting to see the emergence of image generation models trained on public domain material such as image galleries shared by museums, and artworks volunteered by their creators via an opt-in system.

▲ **Prompt:** "an impressionist landscape in the style of Turner. shafts of sunlight are visible through clouds, illuminating the countryside. trees, river". **Negative prompt:** "people, animals"

> IMAGE GENERATION IS A MINEFIELD IF YOU'RE HOPING TO KEEP YOUR USE OF MACHINE LEARNING ETHICAL AND COST-EFFECTIVE

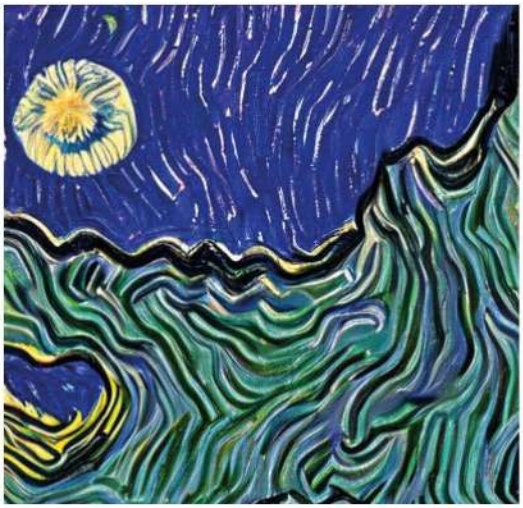
For this tutorial, we've chosen Mitsua Diffusion One (**magpi.cc/mitsua**), a Stable Diffusion-compatible text-to-image model trained exclusively on public domain content. Its licence requires that you don't breach anyone's copyright in fine-tuning the model, and that you don't pass off its generated images as non-AI creations.

Hugging Face's Diffusers library allows us to run diffusion models – which work by reducing noise until an image that resembles their prompt appears – on almost any CPU. We're using Raspberry Pi 5.

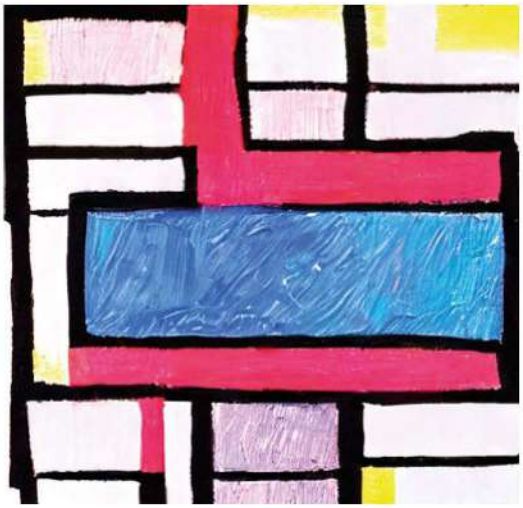
▲ **Prompt:** "ukiyo-e woodblock print of mount fuji, capped with snow, in the style of Hokusai". **Negative prompt:** "people, animals"

▼ **Prompt:** "a painting of ancient japanese kyoto huge temple, kiyomizu temple, an impressionism painting, fine art". **Negative prompt:** "photo"

▼ **Prompt:** "a rainy British street, fine art, John Atkinson Grimshaw, autumn, trees, evening". **Negative prompt:** people, animals



▲ **Prompt:** "a night sky in the style of Van Gogh. The houses of a town are visible beneath



▲ **Prompt:** "an abstract painting in the style of Mondrian, with black lines marking square and rectangular shapes on a white background, some of them filled with red, blue and black colour. oil painting". **Negative prompt:** "humans, animals, curves"

01 Set up your disk

You'll need an 8GB Raspberry Pi 5 (or Raspberry Pi 4) with a 64GB microSD card. On a fresh install of Bookworm – we'll be running headless, so you can either disable booting to GUI or install the Lite version of Raspberry Pi OS. Before you burn your OS image to a microSD card in Raspberry Pi Imager, remember to enable SSH with password authentication, set a hostname, username and password and configure your wireless LAN if you need it.

If you've installed the full GUI-enabled version of Raspberry Pi OS, you'll want to start by disabling that. SSH in or open a terminal and type:

```
$ sudo raspi-config
1 System Options > S5 Boot / Auto Login > B2
Console Autologin
```

Select Finish and say Yes to rebooting

02 Configure swap

If you don't have at least 8GB of swap space – extra working space on your microSD card for when you run out of RAM – Diffusers will fail to output the image after processing. Note that this can wear out your microSD card faster than you'd otherwise expect.

```
$ sudo dphys-swapfile swapoff
$ sudo nano /etc/dphys-swapfile
```

In nano, we'll edit the swapsize and the maxswap entries to give us 8GB of swap and no limit on swap size:

```
CONF_SWAPSIZE=8192
CONF_MAXSWAP=
```

Hit **CTRL** + **X** and **Y**. Then, back at the command prompt:

```
$ sudo dphys-swapfile setup
$ sudo dphys-swapfile swapon
$ reboot
```

03 Large-scale cloning

Now, a few dependencies before we get started. First, we'll need to add large file support to git:

```
$ curl -s https://packagecloud.io/install/
repositories/github/git-lfs/script.deb.sh |
sudo bash
$ sudo apt install git-lfs
```

Initialise it:

```
$ git-lfs install
```

Create a home for our model and download it:

```
$ mkdir -p Software/models && cd Software/
models
$ git clone https://huggingface.co/Mitsua/
mitsua-diffusion-one
```

While not particularly large by the standards of diffusion models, Mitsua Diffusion One still clocks in at around 6GB, so this operation might take a while.

04 You've gotta keep 'em separated

To use Hugging Face Diffusers, you'll need PyTorch, which requires a maximum version of Python 3.10.6 – rather older than the current version available in repos. We'll use pyenv to install it, and the virtualenv command to ensure that we lock our working environment to the right version of Python and keep all the relevant dependencies siloed. This ensures that nothing we do here will have an impact on anything else we have installed or our system-wide software defaults.

```
$ cd
```

```
$ curl https://pyenv.run | bash
```

You'll be prompted to add shortcuts to your bash profiles to invoke pyenv from the command line.

05 Update your profile

In terminal, enter:

```
$ nano ~/.bashrc
```

Add the following block to the bottom of the file, and save with **CTRL**+**X**.

```
echo 'export PYENV_ROOT="$HOME/.pyenv"' >>
~/.bashrc
echo 'command -v pyenv >/dev/null || export
PATH="$PYENV_ROOT/bin:$PATH"' >> ~/.bashrc
echo 'eval "$(pyenv init -)"' >> ~/.bashrc
```

Now edit the .profile file:

```
$ nano ~/.profile
```

As above, add this block to the bottom of the file, and save:

```
echo 'export PYENV_ROOT="$HOME/.pyenv"' >>
~/.profile
echo 'command -v pyenv >/dev/null || export
PATH="$PYENV_ROOT/bin:$PATH"' >> ~/.profile
echo 'eval "$(pyenv init -)"' >> ~/.profile
```

Now reload your bash profile, thus:

```
$source ~/.bashrc
```

06 Python depends on you

We're going to use pyvenv to install the right version of Python, but we have some dependencies to install first. We've split these into three comments for ease of copying, but you can string them into a single apt install command if you prefer.

```
$ sudo apt install zlib1g zlib1g-dev libssl-dev
libbz2-dev libsqlite3-dev
$ sudo apt install libedit-dev libncurses5-dev
liblzma-dev
$ sudo apt-get install libreadline-dev libffi-dev
```

And now we can install the correct Python version, corralled by pyenv.

```
$ pyenv install 3.10.6
```

07 Virtual environment

As another important step to maintaining an air gap between multiple Python setups, we're going to create a virtual environment, in which we'll install

```
$ pyenv virtualenv 3.10.6 venv_diffusion
```

Now we'll activate it. You'll have to repeat this step every time you want to use our diffusion script.

```
$ pyenv activate venv_diffusion
```

The command prompt will now be tagged (`venv_diffusion`) We'll install the software we need to invoke our model.

```
$ cd Software
$ git clone https://codeberg.org/MightyOwlbear/
Raspberry-Pi-5-Diffusion.git
$ cd Raspberry-Pi-5-Diffusion
$ nano python mitsua_app.py
```

08 Nearly there

Enter:

```
$ cd Software
$ git clone https://codeberg.org/MightyOwlbear/
Raspberry-Pi-5-Diffusion.git
$ cd Raspberry-Pi-5-Diffusion
$ nano python mitsua_app.py
```

Top Tip

Don't forget to activate

Type `pyenv activate venv_diffusion` to activate your virtual environment.

On line 9, check to make sure the path matches the one you downloaded Mitsua Diffusion One model to. In our sample script, we've worked with the username 'pi' and otherwise used the directory structure described in this tutorial. Press **CTRL+X** to quit and save when prompted.

```
$ python mitsua_app.py
```

You'll be asked to supply a prompt for the model: a description of the image you'd like it to produce. You'll then be asked for negative prompts: anything you don't want the image to include, from an object, to a colour, to a trait such as blurriness or fused fingers, or a style, such as a photo.

The first time you run this script will take the longest. If everything's configured correctly, however, a standard run should take under 15 minutes, and will produce a time-stamped output file.

09 Line, please

As it is trained on public domain material, primarily from museums and art galleries, plus Wikimedia Commons and public domain subsets of various image sharing sites, Mitsua Diffusion One has specific strengths and weaknesses.

You'll have more luck getting it to produce somewhat convincing images of woodblock prints, engravings, impressionism, and fine art paintings than photographs. We never got it to give us a really convincing picture of a cat.

To get the best results, we've found it useful to name specific artists and styles. As large parts of its training data come from art gallery collections, it's worth browsing, for example, the Art Institute of Chicago's collection (**magpi.cc/aicpub**) to find some inspiration.

Illustrating this article, you'll find examples of prompts and the images they produced.

Generate image descriptions

twinery.org

Image description is incredibly important on the modern internet. Image alt text descriptions are, first and foremost, essential for blind and partially-sighted internet users, but also for anyone interacting with the web using text, including RSS feeds and text browsers like Lynx.

While hand-crafted alt text is better than something automated, not everyone finds it easy to write. And sometimes you might need a tool to quickly describe an image to you for your own reasons. This project uses the same Python venv and directory structure we set up for our copyright-safe image generation project, and uses Salesforce's BLIP (Bootstrapping Language-Image Pre-training) image description model (**magpi.cc/blip**).

If you prefer not to use machine learning tools from tech giants, Salesforce might be one you'd rather avoid. A paper describing BLIP's development (**magpi.cc/blippdf**) indicates that its training is based on image and alt text pairs scraped from the web, which are then filtered to avoid 'noisy' captions. The filtering process and very limited length of the generated captions make the likelihood of generating copyright-infringing content minimal, but the training methodology should be taken into account if you intend on using captions generated by BLIP.

We'll start by downloading the model and image description script.

```
$ cd ~/Software/models
$ git-lfs install
$ git clone https://huggingface.co/
Salesforce/blip-image-captioning-base
$ cd ..
$ git clone https://codeberg.org/
MightyOwlbear/blip-image-description-tool.git
$ cd blip-image-description-tool
$ nano captioner.py
```

Check the paths to make sure they match the location of your model - if your user isn't called 'pi', you'll need to change this. Press **CTRL + X** to save and exit nano.

```
$ pyenv activate venv_diffusion
$ python captioner.py
```

This version of the captioner only accepts web addresses for its image targets. To test it, you can use the following image URL: **magpi.cc/gullsjpg**. This should produce a description that reads: "a group of birds flying in the sky".

mitsua-app.py

DOWNLOAD THE FULL CODE: magpi.cc/pi5diffusion

Language: **Python**

```
001.  from diffusers import StableDiffusionPipeline
002.  from PIL import Image
003.  import datetime
004.
005.  prompt = input("Describe the image you want: ")
006.  negative_prompt = input("Describe traits to avoid - e.g.
      blurry, fused hands: ")
007.
008.  # insert path to Mitsua Diffusion One below
009.  pipe = StableDiffusionPipeline.from_pretrained("/home/pi/
      Software/models/mitsua-diffusion-one/",
      low_cpu_mem_usage=True)
010.  pipe = pipe.to("cpu")
011.
012.  image = pipe(prompt, negative_prompt=negative_prompt,
      num_inference_steps=31, width=400, height=400).images[0]
013.
014.  # timecode every output image so they can't overwrite each
      other
015.  created = datetime.datetime.now().strftime("%Y%m%d%H%M%S")
016.  image.save("output-" + created + ".png")
```

Top Tip

The htop

If you have a screen connected to Raspberry Pi, you can run `htop` to monitor memory and swap consumption.

captioner.py

DOWNLOAD THE FULL CODE: magpi.cc/bliptool

Language: **Python**

```
001.  import requests
002.  from PIL import Image
003.  from transformers import BlipProcessor,
      BlipForConditionalGeneration
004.
005.  # insert the path to your model's directory in both of the
      following lines
006.  processor = BlipProcessor.from_pretrained("/home/pi/Software/
      models/blip-image-captioning-base/")
007.  model = BlipForConditionalGeneration.from_pretrained("/home/
      pi/Software/models/blip-image-captioning-base/")
008.
009.  img_url = input("Enter JPG image URL: ")
010.  raw_image = Image.open(
      requests.get(img_url, stream=True).raw).convert('RGB')
011.
012.  inputs = processor(raw_image, return_tensors="pt")
013.
014.  out = model.generate(**inputs, max_new_tokens=1000)
015.  print(processor.decode(out[0], skip_special_tokens=True))
```

Part 03

Using your Raspberry Pi

Learn about the Raspberry Pi operating system

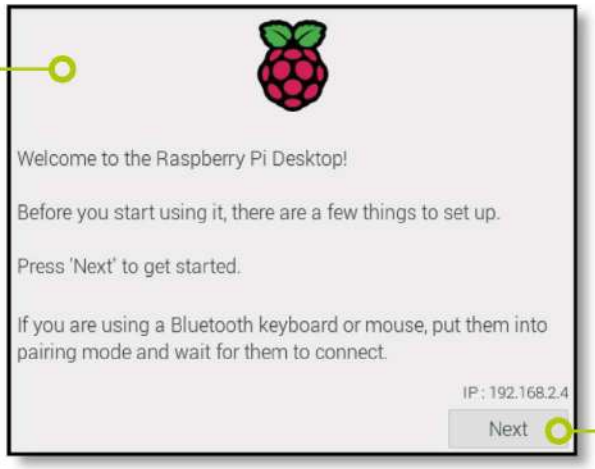


Figure 1. The Welcome Wizard

MAKER

Gareth Halfacree

Gareth is a freelance technology journalist, writer, and former system administrator in the education sector with a passion for open-source software and hardware.

freelance.halfacree.co.uk

Your Raspberry Pi can run a wide range of software, including a number of different operating systems – the core software that makes a computer run. The most popular of these, and the official operating system of Raspberry Pi, is Raspberry Pi OS. Based on Debian Linux, it is tailor-made for Raspberry Pi and comes with a range of extra software pre-installed and ready to go.

If you've only ever used Microsoft Windows or Apple macOS, don't worry: Raspberry Pi OS is based on the same intuitive windows, icons, menus, and pointer (WIMP) principles, and should quickly become familiar.

The first time you run Raspberry Pi OS, you'll see the Welcome Wizard (**Figure 1**). This helpful tool will walk you through changing some settings in Raspberry Pi OS, known as the configuration, to match how and where you will be using your Raspberry Pi.

Click the Next button, then choose your country, language, and time zone by clicking on each drop-down box in turn and selecting your answer

▲ **Figure 2:** Selecting a language, among other options

from the list (**Figure 2**). If you are using a US-layout keyboard, click on the check box to make sure Raspberry Pi OS uses the correct keyboard layout. If you want the desktop and programs to appear in English, regardless of your country's native language, click on the Use English language checkbox to tick it. When you're finished, click Next.

▼ **Figure 3:** Setting a new password

The next screen will ask you to choose a name and password for your user account (**Figure 3**). Choose a name – it can be anything you like, but it must start with a letter and can only contain lower-case letters, digits, and hyphens. Then you'll need to create a memorable password. You'll be asked to type the password twice to make sure you didn't make any mistakes that could lock you out of your new account. When you're happy with your choices, click Next.

The following screen will allow you to choose your Wi-Fi network from a list (**Figure 4**).

Scroll through the list of networks with the mouse or keyboard, find your network's name, click on it, then click Next. Assuming that your wireless network is secure (it really should be), you'll be asked for its password (also known as its pre-shared key). If you don't use a custom password, the default is normally written on a card that comes with the router, or on the bottom or back of the router itself. Click Next to connect

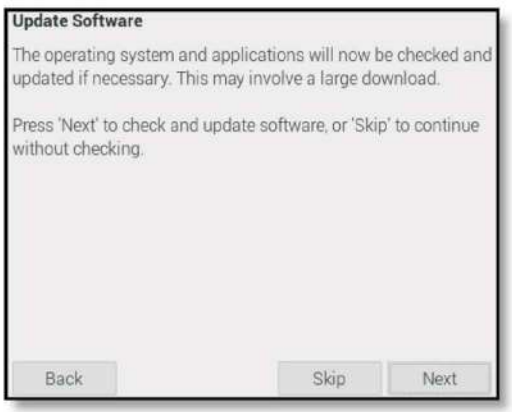
▲ **Figure 4:** Choosing a wireless network

▼ **Figure 5:** Selecting a browser

to the network. If you don't want to connect to a wireless network, click Skip.

Next, you will be asked to choose your default web browser from the two preinstalled in Raspberry Pi OS: Google's Chromium, the default, and Mozilla's Firefox (**Figure 5**). For now, just leave Chromium selected as the default, so you can follow along with this book; you can always switch to Firefox later, if you'd prefer. If you do change the default browser, you can also choose to uninstall the non-default browser to save space on your microSD card. Just tick the box when you are offered the option, and click the Next button.

The next screen will allow you to check for and install updates for Raspberry Pi OS and the other software on Raspberry Pi (**Figure 6**). Raspberry Pi OS is regularly updated to fix bugs, add new features, and improve performance. To install these updates, click Next. Otherwise, click Skip. Downloading the updates can take several minutes, so be patient.



▲ **Figure 6:** Checking for updates

You'll Need

- Raspberry Pi
- Raspberry Pi OS

Warning!
Power supply

If, after your Raspberry Pi starts up, you see a message in the top-right corner telling you "this power supply is not capable of supplying 5A." it means you're using a power supply which can't supply the 5V at 5A required by Raspberry Pi 5. You should replace your power supply with one supporting Raspberry Pi 5.

magpi.cc/power

Top Tip

Wireless networking

Built-in wireless networking is only available on Raspberry Pi 3, Raspberry Pi 4, Raspberry Pi 5, and Raspberry Pi Zero W and Zero 2 W families. If you want to use a different model of Raspberry Pi with a wireless network, you'll need a USB Wi-Fi adapter.

Figure 7: Restarting Raspberry Pi

When the updates are installed, a window saying 'System is up to date' will appear; click the OK button.

The final screen of the Welcome Wizard (**Figure 7**) provides one last bit of information: certain changes made will only take effect when you restart your Raspberry Pi (a process also known as rebooting). Click the Restart button and your Raspberry Pi will restart. From now on, the Welcome Wizard won't appear; its job is done, and your Raspberry Pi is ready to use.

Navigating the desktop

The version of Raspberry Pi OS installed on most Raspberry Pi boards is properly known as 'Raspberry Pi OS with desktop', referring to its main graphical user interface (**Figure 8**). The bulk of this desktop is taken up with a wallpaper picture (**A** in **Figure 8**), on top of which the programs you run will appear. At the top of the desktop is a taskbar (**B**), which allows you to launch your installed programs. These are then indicated by tasks (**C**) in the taskbar.

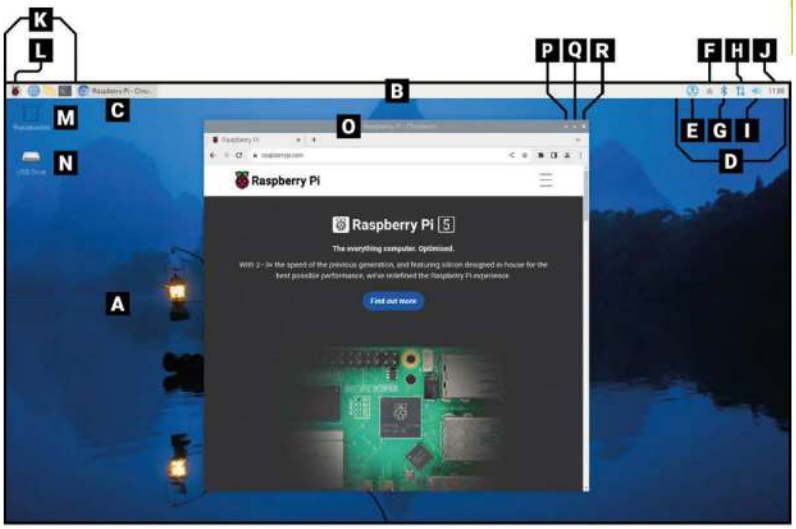


Figure 8: The Raspberry Pi OS desktop

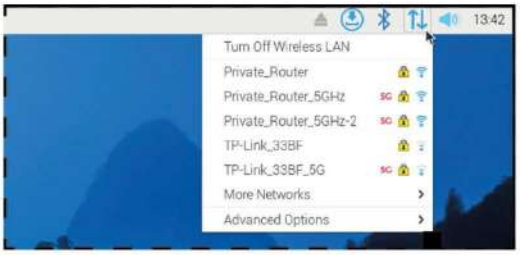
A. Wallpaper
B. Taskbar
C. Task
D. System Tray
E. Software Update icon
F. Media eject
G. Bluetooth icon
H. Network icon
I. Volume icon
J. Clock
K. Launcher
L. Menu (or Raspberry Pi icon)
M. Wastebasket icon
N. Removable drive icon
O. Window title bar
P. Minimise
Q. Maximise
R. Close

The right-hand side of the menu bar houses the system tray (**D**). The software update icon (**E**) appears only when there are updates to Raspberry Pi OS and its applications. If you have any removable storage, such as USB memory sticks, connected to Raspberry Pi you'll see an eject symbol (**F**); click this to safely eject and remove them. On the far right is the clock (**J**); click it to bring up a digital calendar (**Figure 9**).

Next to this is a speaker icon (**I** in **Figure 8**). Click on it with the left mouse button to adjust audio volume, or click with the right mouse button to choose which output Raspberry Pi should use for its sound. Next to that is a network icon (**H**); if you're connected to a wireless network you'll see the signal strength displayed as a series of bars, but if you're connected to a wired network you'll just see two arrows. Clicking the network icon will bring up a list of nearby wireless networks (**Figure 10**), while clicking on the Bluetooth icon (**G**) next to that will allow you to connect to a nearby Bluetooth device.

The left-hand side of the menu bar is home to the launcher (**K**), which is where you'll find the programs installed alongside Raspberry Pi OS.

Figure 9: The digital calendar

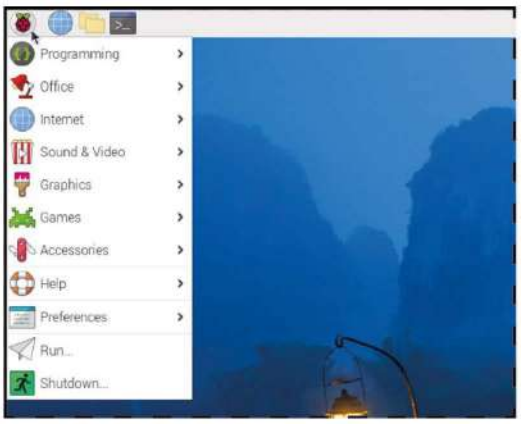


▲ **Figure 10:** Listing nearby wireless networks

Some of these are visible as shortcut icons; others are hidden away in the menu, which you can bring up by clicking the Raspberry Pi icon (**L**) to the far left (**Figure 11**).

The programs in the menu are split into categories. Each category's name tells you what to expect: the Programming category contains software designed to help you write your own programs while Games will help you while away the hours.

Not every program will be detailed in this guide, so feel free to experiment with them to learn more. On the desktop, you'll find the Wastebasket (**M**) and any external storage devices (**N**) connected to your Raspberry Pi.



▲ **Figure 11:** The Raspberry Pi menu

The Chromium web browser

To practice using your Raspberry Pi, start by loading the Chromium web browser: click on the Raspberry Pi icon at the top-left to bring up the menu, move your mouse pointer to select the Internet category, and click on Chromium Web Browser to load it.

If you've used Google's Chrome browser on another computer, the Chromium browser will be immediately familiar. Chromium lets you visit websites, play videos, games, and even communicate with people all over the world on forums and chat sites.

> "The Programming category contains software to write your own programs"

Start using Chromium by maximising its window so it fills the screen: find the three icons at the top-right of the Chromium window title bar (**O**) and click on the middle, up-arrow icon (**Q**). This is the maximise button. To the left of maximise is minimise (**P**), which will hide a window until you click on it in the taskbar at the top of the screen. The cross to the right of maximise is close (**R**), which does exactly what you'd expect: it closes the window.

The first time you run the Chromium web browser, the Raspberry Pi website should load automatically, as shown in **Figure 12**. If not (or to visit other websites), click in the address bar at the top of the Chromium window – the big white bar with a magnifying glass on the left-hand side – and type **raspberrypi.com** (or the address of the website you want to visit), then press the **ENTER** key on your keyboard. The Raspberry Pi website will load.

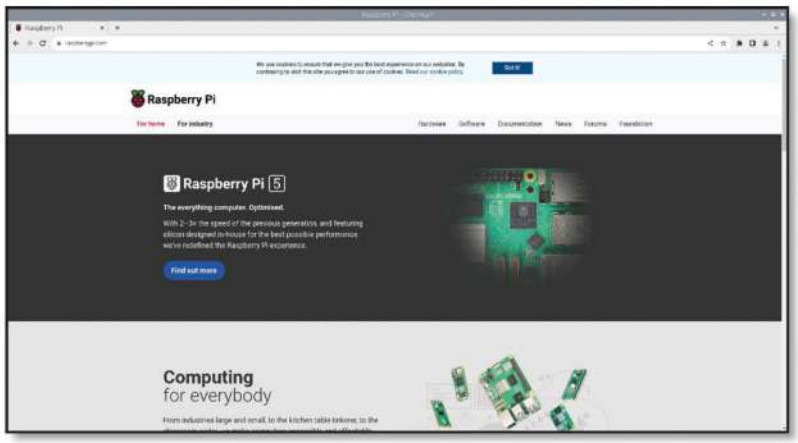
You can also type searches into the address bar: try searching for 'Raspberry Pi', 'Raspberry Pi OS', or 'retro gaming'.

The first time you load Chromium, it may bring up several tabs along the top of the window. To switch to a different tab, click on it; to close a tab without closing Chromium itself, click the cross on the right-hand side of the tab you want to close.

Top Tip

Shutdown safely

Never remove the power cable from a Raspberry Pi or turn the power supply off at the wall without shutting down first. Doing so is likely to corrupt the operating system, and you could also lose any files you have created or downloaded.



▼ **Figure 12:** The Raspberry Pi website in Chromium

To open a new tab, which is a handy way of having multiple websites open without having to juggle multiple Chromium windows, either click on the tab button to the right of the last tab in the list, or hold down the **CTRL** key on the keyboard and press the T key before letting go of **CTRL**.

When you're finished with Chromium, click the close button at the top-right of the window.

Top Tip

Close and save

Closing a window before you've saved any work you've done is a bad idea; while many programs will warn you to save when you click the close button, others won't.

The File Manager

Files you save – for example, programs, videos, images – all go into your home directory. To see the home directory, click on the Raspberry Pi icon again to bring up the menu, move the mouse pointer to select Accessories, then click on File Manager to load it (**Figure 13**).

The File Manager lets you browse the files and folders, also known as directories, on Raspberry Pi's microSD card, as well as those on any removable storage devices – like USB flash drives – you have connected to your Raspberry Pi's USB ports. When you first open it, it automatically goes to your home directory. In here you'll find a series of other folders, known as subdirectories, which – like the menu – are arranged in categories. The main subdirectories are:

1. **Bookshelf** – This contains digital copies of books and magazines from Raspberry Pi Press. You can read and download books with the Bookshelf application in the Help section of the menu.

2. **Desktop** – This folder is what you see when you first load Raspberry Pi OS. If you save a file in here it will appear on the desktop, making it easy to find and load.

3. **Documents** – Home to most of the text files you'll create, from short stories to recipes.

4. **Downloads** – When you download a file from the internet using the Chromium web browser, it will be automatically saved in Downloads.

5. **Music** – Any music you create or download can be stored here.

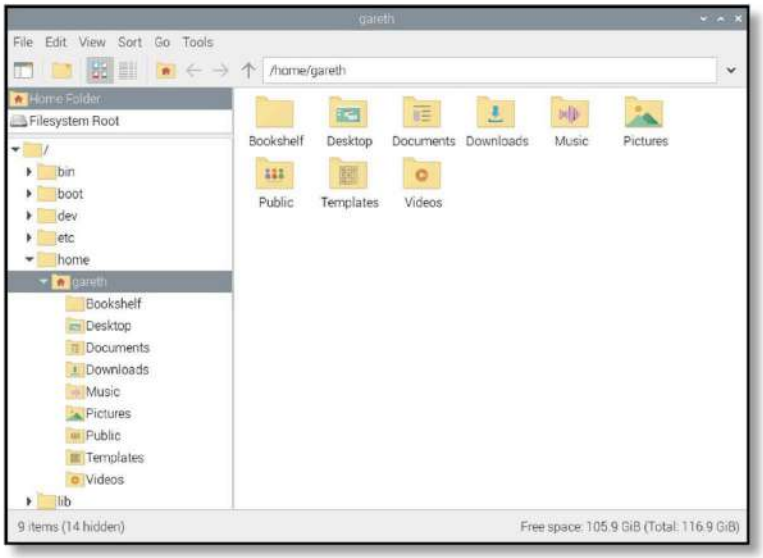
6. **Pictures** – This folder is specifically for pictures, known in technical terms as image files.

7. **Public** – While most of your files are private, anything you put in Public will be available to other users of your Raspberry Pi, even if they have their own username and password.

8. **Templates** – This folder contains any templates – blank documents with a basic layout or structure already in place – which have been installed by your applications or created by you.

9. **Videos** – A folder for videos, and the first place most video-playing programs will check for content.

▼ **Figure 13:** The file manager



▲ **Figure 14:** Inserting a removable storage device

The File Manager window itself is split into two main panes: the left pane shows the directories on your Raspberry Pi, and the right pane shows the files and subdirectories of the directory selected in the left panel.

If you plug a removable storage device into the Raspberry Pi's USB port, a window will pop up asking if you'd like to open it in the File Manager (**Figure 14**). Click OK and you'll be able to see its files and directories.

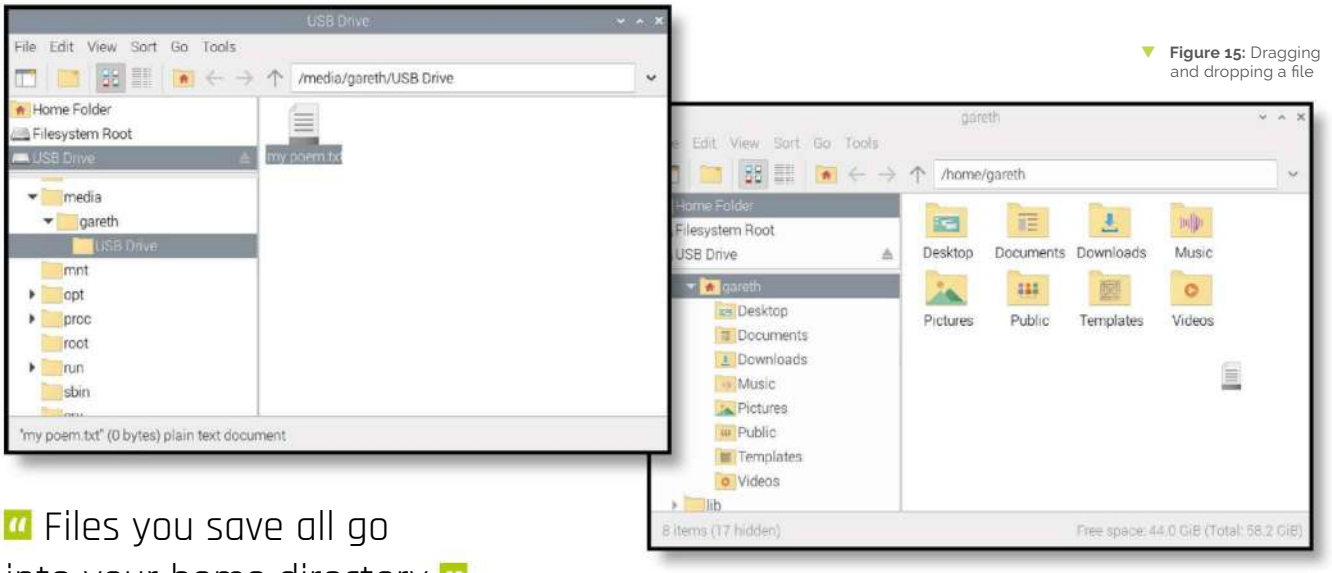


Figure 15: Dragging and dropping a file

> Files you save all go into your home directory

You can easily drag and drop files between Raspberry Pi's microSD card and a removable device. With your home directory and the removable device open in separate File Manager windows, move your mouse pointer to the file you want to copy, click and hold the left mouse button down, slide your mouse pointer to the other window, and let go of the mouse button (**Figure 15**).

An easy way to copy a file is to click once on the file, click the Edit menu, click Copy, click the other window, then click the Edit menu and click Paste.

The Cut option, also available in the Edit menu, is similar, but it deletes the file from its original home after making the copy. Both options can also be used through the keyboard shortcuts **CTRL+C** (copy) or **CTRL+X** (cut), and **CTRL+V** (paste).

When you've finished experimenting, close the File Manager by clicking the close button at the very top-right of the window. If you have more than one window open, close them all. If you connected a removable storage device to your Raspberry Pi, eject it by clicking the eject button at the top-right of the screen, finding it in the list, and clicking on it before unplugging it.

The Recommended Software tool

Raspberry Pi OS comes with a wide range of software already installed, but your Raspberry Pi is compatible with even more. A selection of the best of this software can be found in the Recommended Software tool.

Note that the Recommended Software tool needs a connection to the internet. If your Raspberry Pi is connected, click on the Raspberry Pi icon, move your mouse pointer to Preferences, and click on Recommended Software. The tool will load and start downloading information about available software.

After a few seconds, a list of compatible software packages will appear (**Figure 16**). These, like the software in the Raspberry Pi menu, are arranged into various categories. Click on a category in the pane on the left to see software from that category, or click All Programs to see everything.

If a piece of software has a tick next to it, it's already installed on your Raspberry Pi. If it doesn't, you can click on the check box next to it to add a tick and mark it for installation. You can mark as many pieces of software as you like before installing them all at once, but if you're using a smaller-than-recommended microSD card you may not have room for them all.

Some versions of Raspberry Pi OS come with more software installed than others. If the Recommended Software Tool says Code the Classics is already installed – if there's already a

Figure 16: The Recommended Software tool

Top Tip

Eject devices

Always use the eject button before unplugging an external storage device. If you don't, the files on it may become corrupt and unusable.

Top Tip

Save your work

Get in the habit of saving your work, even if you haven't finished it yet. It will save you a lot of trouble if there's a power cut and you're interrupted part-way through!

tick in the checkbox – you can choose something else from the list to install instead.

There's software available for Raspberry Pi OS to perform a wide range of tasks, including a selection of games written for the book Code the Classics, Volume 1 – a walk through the history of gaming which teaches you how to write your own games in Python, available at **store.rpipress.cc**.

To install the Code the Classics games, click on the checkbox next to Code the Classics to tick it; you may need to scroll down the list of applications to see it. You'll see the text (will be installed) appear to the right of the application you selected, as shown in **Figure 17**.

Click Apply to install the software; you'll be asked to enter your password. It will take up to a minute, depending on the speed of your internet connection, to install (**Figure 18**). Once the process is finished, you'll see a message telling you that installation is complete. Click OK to close the dialogue box, then click the Close button to close the Recommended Software tool.

Figure 17: Selecting Code the Classics for installation

Figure 18: Installing Code the Classics

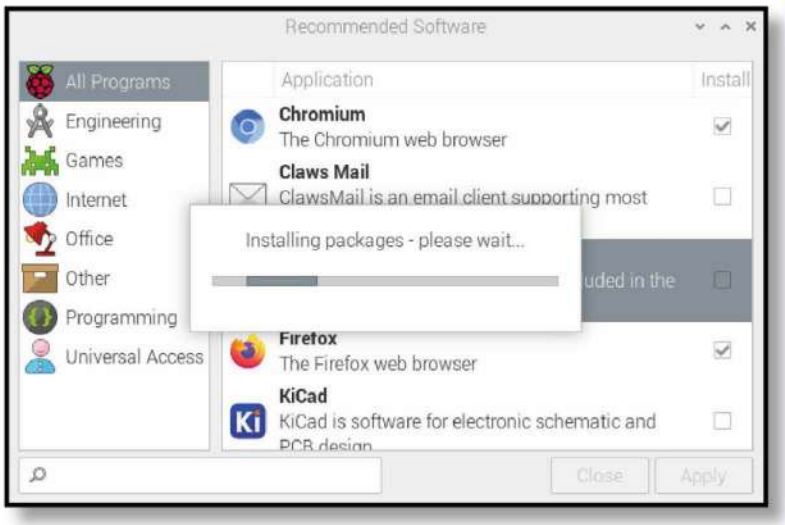
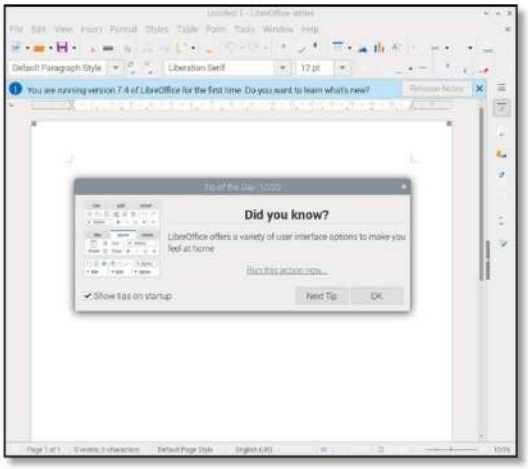


Figure 19: The LibreOffice Writer program



If you change your mind about software you've installed, you can free up space by uninstalling it. Just load the Recommended Software tool again, find the software in the list, and click the checkbox to remove the tick. When you click Apply, the software will be removed, but any files you've created with it and saved in your Documents folder will remain.

Another tool for installing or uninstalling software, the Add/Remove Software tool, can be found in the same Preferences category of the Raspberry Pi menu. This offers a wider selection of software beyond the list of recommended software.

The LibreOffice productivity suite

For another taste of what Raspberry Pi can do, click on the Raspberry Pi icon, move your mouse pointer to Office, and click on LibreOffice Writer. This will load the word processor portion of LibreOffice (**Figure 19**), a popular open-source productivity suite.

If you don't have an Office category in your Raspberry Pi menu, or if you can't find LibreOffice Writer in there, it may not be installed. Go back to the Recommended Software tool and install it there before proceeding with this section.

A word processor lets you write and format documents: you can change the font style, colour, size, add effects, and even insert pictures, charts, tables, and other content. A word processor also lets you check your work for mistakes, highlighting spelling and grammar problems in red and green respectively as you type.

Begin by writing a paragraph so you can experiment with formatting. If you're feeling particularly keen, you could write about what you've learned about Raspberry Pi and its software so far. Explore the different icons at the top of the window to see what they do: see if you can make your writing bigger and change its colour. If you're not sure how to do this, simply move your mouse pointer over each icon to display a 'tool tip' telling you what that icon does. When you're satisfied, click the File menu and the Save option to save your work (**Figure 20**). Give it a name and click the Save button.

LibreOffice Writer is only part of the overall LibreOffice productivity suite. The other parts, which you'll find in the same Office menu category as LibreOffice Writer, are:

10. **LibreOffice Base** – A database: a tool for storing information, looking it up quickly, and analysing it.
11. **LibreOffice Calc** – A spreadsheet: a tool for handling numbers and creating charts and graphs.
12. **LibreOffice Draw** – An illustration program: a tool for creating pictures and diagrams.
13. **LibreOffice Impress** – A presentation program: for creating slides and running slideshows.
14. **LibreOffice Math** – A formula editor: for creating properly-formatted mathematical formulae which can be used in other documents.

LibreOffice is also available for other computers and operating systems. If you enjoy using it on your Raspberry Pi, you can download it for free from libreoffice.org and install it on any Microsoft Windows, Apple macOS, or Linux computer. You can close LibreOffice Writer by clicking the close button at the top-right of the window.

Top Tip

Getting help

Most programs include a Help menu which has everything from information about what the program is to guides on how to use it. If you ever feel lost or overwhelmed by a program, look for the Help menu to reorient yourself.

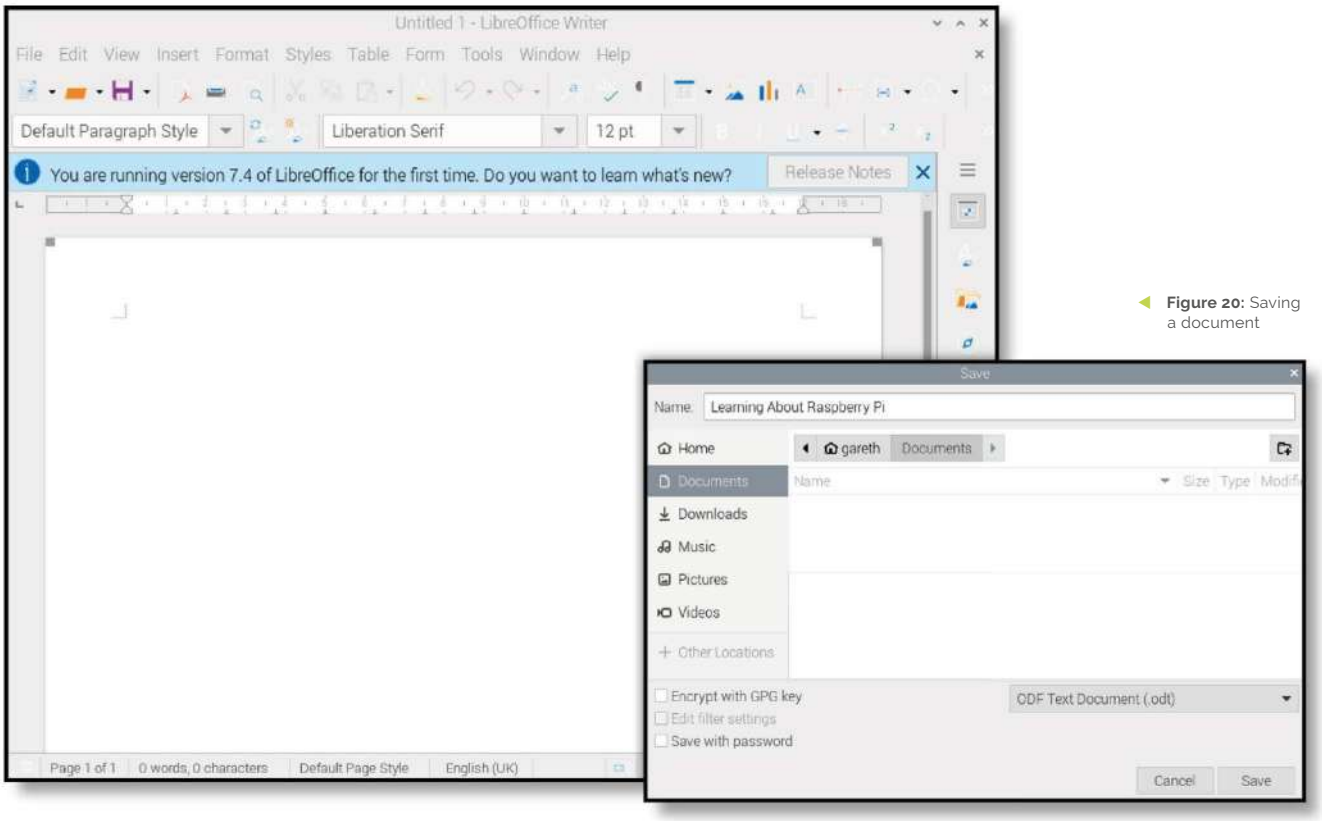


Figure 20: Saving a document

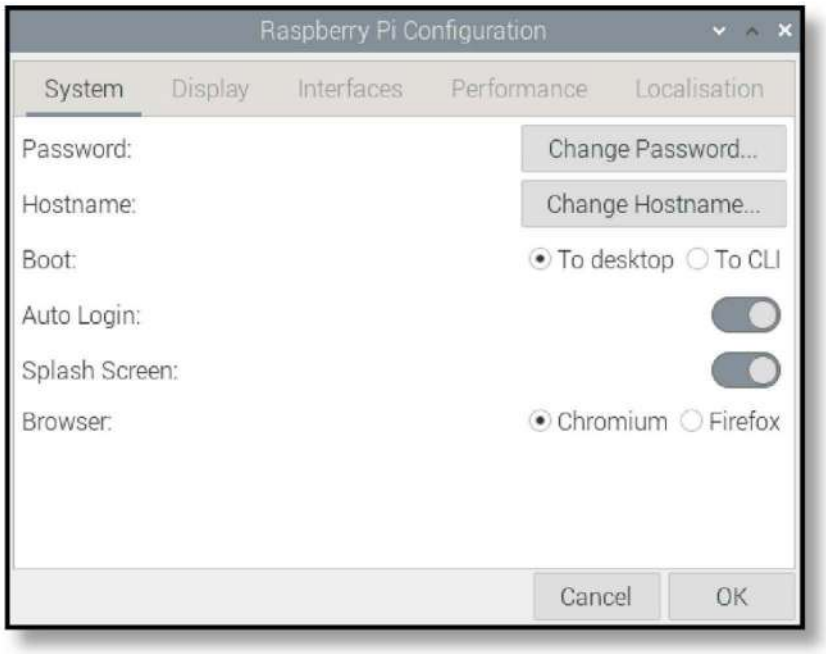


Figure 21: The Raspberry Pi Configuration tool

Raspberry Pi Configuration tool

The last program you'll learn about in this chapter is known as the Raspberry Pi Configuration tool, and it's a lot like the Welcome Wizard you used at the start: it allows you to change various settings in Raspberry Pi OS. Click on the Raspberry Pi icon, move your mouse pointer to select the Preferences category, then click on Raspberry Pi Configuration to load it (**Figure 21**).

The tool is split into five tabs. The first of these is System: this allows you to change the password of your account, set a host name – the name your Raspberry Pi uses on your local wireless or wired network – and alter a range of other settings, including choosing a default web browser. The majority of these shouldn't need changing. Click on the Display tab to bring up the next category. Here you can alter the screen display settings, if needed, to suit your TV or monitor.

The Interfaces tab offers a range of settings, all of which (except for Serial Console and Serial Port) start off disabled. These settings should only be changed if you're adding new hardware, and even then only if instructed by the hardware's manufacturer. The exceptions to this rule are SSH, which enables a 'Secure Shell' and lets you log into Raspberry Pi from another computer on your network using an SSH client; VNC, which enables a 'Virtual Network Computer' and lets you see and control the Raspberry Pi OS desktop from another computer on your network using a VNC client; and Remote GPIO, which lets you use Raspberry Pi's GPIO pins.

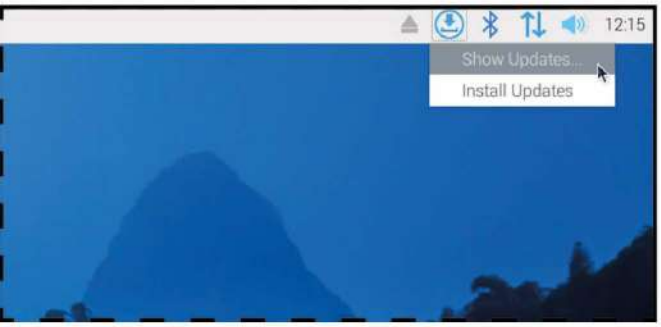
Click on the Performance tab to see the fourth category. Here you configure the overlay file system, which is a way to run your Raspberry Pi without writing changes to the microSD card. It's not something you'll need to do in most cases, so most users can just leave this section as-is.

Finally, click on the Localisation tab to see the last category. Here you can change your locale, which controls things like the language used in Raspberry Pi OS and how numbers are displayed; change the time zone; change the keyboard layout; and set your country for Wi-Fi purposes. For now, though, just click on Cancel to close the tool without making any changes.

Software updates

Raspberry Pi OS receives frequent updates, which add new features or fix bugs. If Raspberry Pi is connected to a network via an Ethernet cable or Wi-Fi, it will automatically check for updates and let you know if any are ready to be installed with a small icon in the system tray (it looks like an arrow pointing down into a tray, surrounded by a circle).

If you see this icon at the top-right of your desktop, there are updates ready to install. Click the icon then click Install Updates to download and install them. If you'd prefer to see what the updates are first, click Show Updates to see a list (**Figure 22**).



◀ **Figure 22:** Using the software update tool

▼ **Figure 23:** Shutting down Raspberry Pi

The time it takes to install updates varies depending on how many there are and how fast your internet connection is, but it should only take a few minutes. After the updates are installed, the icon will disappear from the system tray until there are more updates to install.

Some updates are designed to improve the security of Raspberry Pi OS. It's important to use the software update tool to keep your operating system up-to-date!

Shutting down

Now you've explored the Raspberry Pi OS desktop, it's time to learn a very important skill: safely shutting your Raspberry Pi down. Like any computer, Raspberry Pi keeps the files you're working on in volatile memory – memory which is emptied when the system is switched off. For documents you're creating, it's enough to save each in turn – which moves the file from volatile memory to non-volatile memory (the microSD card) – to ensure you don't lose anything.

The documents you're working on aren't the only files open, though. Raspberry Pi OS itself keeps a number of files open while it's running, and pulling the power cable from your Raspberry Pi while these are still open can result in the operating system becoming corrupt and needing to be reinstalled.

To prevent this from happening, you need to make sure you tell Raspberry Pi OS to save all its files and prepare to be powered off – a process known as shutting down the operating system.

Click on the Raspberry Pi icon at the top left of the desktop and then click on Shutdown. A window will appear with three options (**Figure 23**): Shutdown, Reboot, and Logout. Shutdown is the option you'll use most: clicking on this will tell Raspberry Pi OS to close all open software and files, then shut the Raspberry Pi down. Once the display has gone black, wait a few seconds until the flashing green light on your Raspberry Pi goes off, after which it's safe to turn off the power supply.

If you press the button once, you'll see the same window appear as if you'd clicked the Raspberry Pi icon followed by Shutdown; press the power button again when the window is visible and Raspberry Pi will shut down safely.

If you press and hold the power button for longer, it will perform a hard shutdown – effectively the same as if you'd just turned the power off. Only do this if your Raspberry Pi isn't responding to your instructions and you can't shut down any other way, as it runs the risk of corrupting your files or operating system.

To turn Raspberry Pi back on disconnect then reconnect the power cable, or toggle the power at the wall socket.

Reboot goes through a similar process to Shutdown, closing everything down, but instead of turning Raspberry Pi's power off, it restarts Raspberry Pi – as if you'd chosen Shutdown, then disconnected and reconnected the power cable. You'll need to use Reboot if you make certain changes which require a restart of the operating system – such as installing certain updates to its core software – or if some software has gone wrong, known as crashing, and left Raspberry Pi OS in an unusable state.

Logout is useful if you have more than one user account on your Raspberry Pi: it closes any programs you currently have open and takes you to a login screen on which you are prompted for a username and password. If you hit Logout by mistake and want to get back in, simply type the username and whatever password you chose in the Welcome Wizard at the start of this feature.

Top Tip

Wi-Fi rules

Different countries have different rules about what frequencies a Wi-Fi radio can use. Setting the Wi-Fi country in the Raspberry Pi Configuration Tool to a different country from the one you're actually in is likely to make it struggle to connect to your networks and can even be illegal under radio licensing laws – so don't do it!

Part 02

Learn Python: explore functions and build a CLI app

Let's get functional with Python. In this tutorial, you'll learn how to define and call functions to save yourself writing the same code over and over again

MAKER
Lucy Hattersley

Lucy is editor of *The MagPi* and she is mostly functional.

magpi.cc

Functions are the beating heart of Python. They help you break down Python programs into small, manageable parts that can be easily repeated. They make your code more modular, and easier to understand.

Some people like to think of functions like cutting and pasting. You write the code once (known as 'defining') and then paste the code (called 'calling'). While this is helpful to start with functions are more powerful than cut-and-paste because they accept input and provide output.

In this tutorial, we'll look at defining and calling functions with different arguments. Then we will use this to create an advanced version of the ToDo program from last issue that saves our list to a file, and can be run from anywhere in the CLI (Command Line Interface).

> Functions are the beating heart of Python

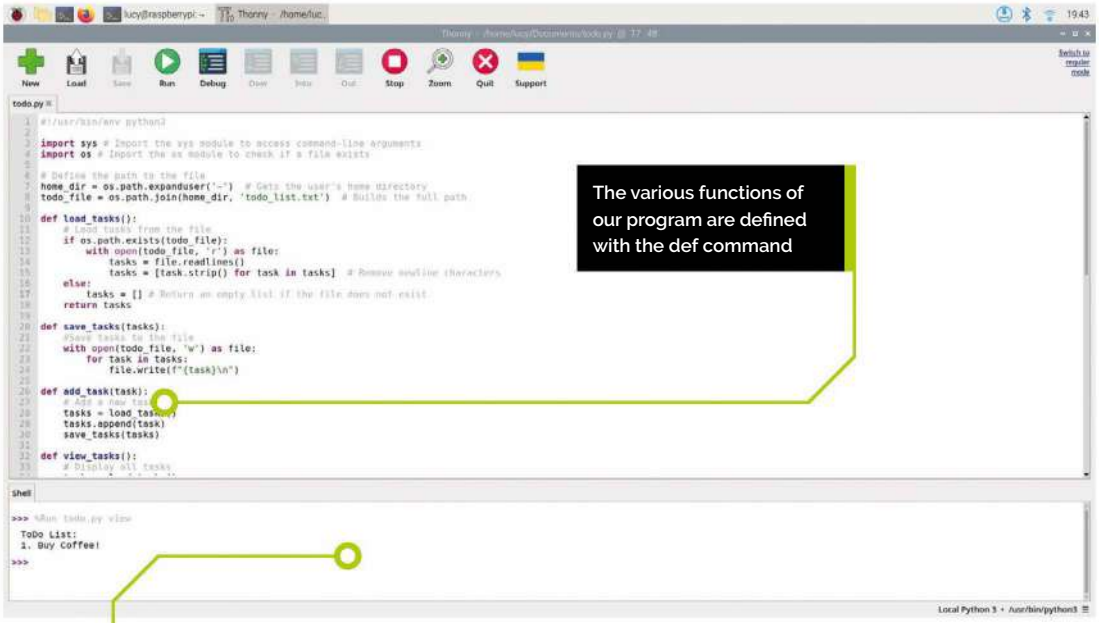
Functions are defined using the `def` command, followed by the name of the function and parentheses (which contain optional parameters are passed into the function). An optional `return` command sends any results back from the function to your main program. It looks like this dummy code:

```
def function(optional_parameter):
        # code to do things
        return optional_return_value
```

The code underneath a function is indented by four spaces. This lets Python know that the indented code is run when the function is used (known as a "function call").

Let's create a function for real. Open the Thonny IDE and create a new file (save it as **add_numbers.py**. Now enter this code:

```
# Define a function that takes two numbers as
arguments and returns their sum
def add_numbers(num1, num2):
    total = num1 + num2
    return total

# Call the function with 2 and 3 as arguments
result = add_numbers(2, 3)
print(f"The sum is: {result}")
```



You'll Need

- Raspberry Pi
- Raspberry Pi OS
- Thonny IDE

The function of this program is to take two numbers and add them together. Pretty simple. The program comes in two parts. The first part is the `def` that defines the function:

```
def add_numbers(num1, num2)
```

The second part is the function call, which asks the `add_numbers` function to add the numbers 2 and 3 together and store the returned `total` variable in another variable called `result`.

```
result = add_numbers(2, 3)
```

Finally, we print out the answer using `print()`. Click Run in Thonny and you should see: "The sum is: 5" in the Shell window below.

REPL to sender

One of the joys of Python is that you can interact with programs in the Shell while running them thanks to REPL (Read-Eval-Print Loop). Click in the Shell and call the function directly with any values you want.

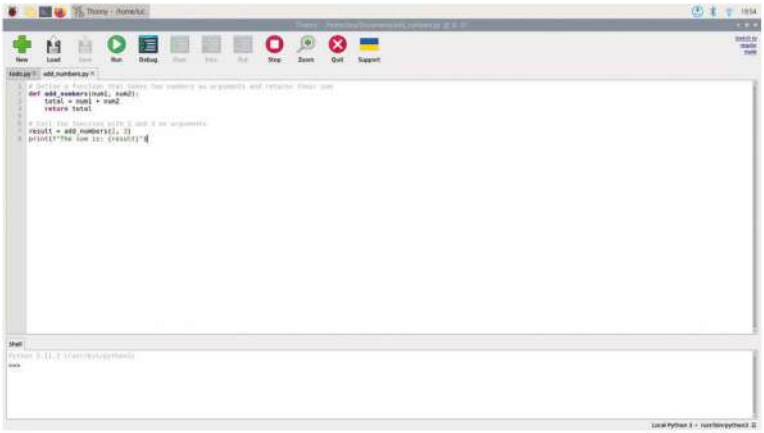
```
add_numbers(6, 2)
```

> The code underneath a function is indented by four spaces

This returns 8 in the Shell. While we're in REPL we should chat about "scope" or "function scope" in this particular concept. In REPL you can access and look at the variables of your program directly. Enter `result` and Shell displays "5" (the value stored by the program on line 7 during the function call when it returns the total value).

Enter `total`, however, and Shell returns:

```
Traceback (most recent call last):
  File "<pyshell>", line 1, in <module>
```



Our introduction to functions takes two numbers and adds them together

```
NameError: name 'total' is not defined
```

Why can we access the `result` variable, but not the `total` variable? The answer is that variables inside function definitions (the part indented under `def`) are "local". That means they can only be accessed locally inside that function.

Once you get back out of the function and into your program the local variables (like our `total`) are no longer needed, and in a modern programming environment like Python the memory space they cleaned and reused.

This is why it is important to use `return` and store any variables or output from your function that you want to use in the rest of your code.

Default variables

Let's raise another problem in Shell: the `add_numbers()` function call without any arguments:

```
add_numbers()
```

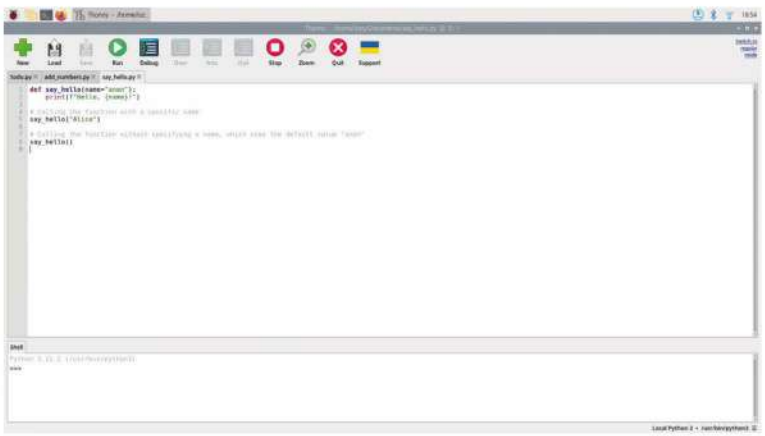
Shell returns another error message:

```
TypeError: add_numbers() missing 2 required
positional arguments: 'num1' and 'num2'
```

Reading the error message tells us what's going wrong. Our "missing 2 required positional arguments" (the two numbers).

We can fix this by adding two numbers to the function call. But it is also possible to add default parameters to a function definition.

Say we are creating a program that addresses people by name, but allows them to remain anonymous. Our function could say "hello" when they enter their name



▼ This function has a default value in case an argument is missing. In this instance, it uses 'anon' instead of a provided name

```
def say_hello(name):
    print(f"Hello, {name}")
```

We could call this function using :

```
say_hello("Alice")
```

But if we want to include the anonymous default we add it to the parameters using `name="anon"`. Enter the code from **say_hello.py**.

```
def say_hello(name="anon"):
    print(f"Hello, {name}!")

# Calling the function with a specific name
say_hello("Alice")

# Calling the function without specifying a
name, which uses the default value "anon"
say_hello()
```

Run this and you will see:

```
Hello, Alice!
Hello, anon!
```

Getting into arguments

One of the limitations of the functions we've created so far is that they only work with a specific number of arguments. Our **add_numbers.py** program only adds two numbers together, for example.

What if want to manage multiple arguments of different amounts? In this case you need to know about the `*args` parameter.

This is a special type of parameter which handles multiple arguments. The "*" tells Python to pack these into a special type of list called a "tuple". Create and Run the following program, calling it **add_mult_numbers.py**.

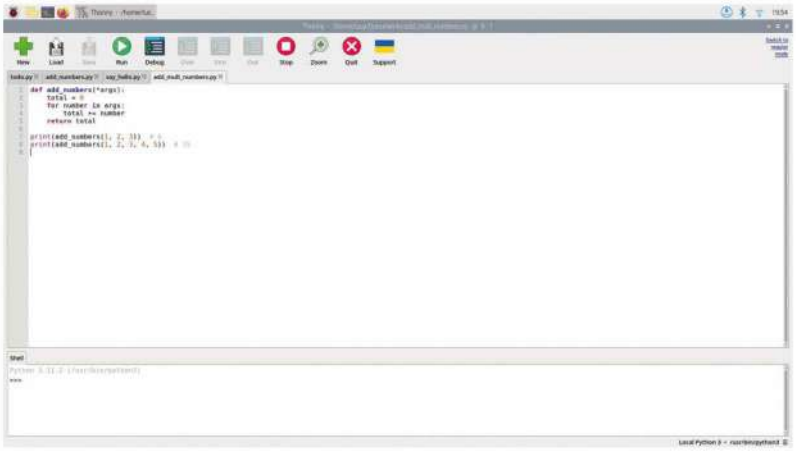
```
def add_numbers(*args):
    total = 0
    for number in args:
        total += number
    return total

print(add_numbers(1, 2, 3))  # 6
print(add_numbers(1, 2, 3, 4, 5))  # 15
```

Run it and Shell returns 6 and 15. There is also a second special parameter called `**kvargs` which

Top Tip

Parameters vs Arguments

Parameters are the variables that are listed inside the parentheses in the function definition. Arguments are the values passed to the function when it is called.



This function takes multiple arguments using the *args parameter instead of specifying individual parameters

handles multiple key/value pairs (which we're not going to go into here as key/value pairs would be a distracting subject). There are two values to unpack hence the two asterisks.

Recursion

Take a deep breath, as this one is likely to hurt your head. Recursion is a relatively hairy concept in programming that quickly causes a headache if you think too hard.

The general concept is that a function definition contains a call to itself. When the function is called, it – in turn – calls another version of itself with a modified argument.

A simple recursive function is this one, that calculates the sum of all numbers up to a given number. For instance, to find the sum of all numbers up to 4, you'd calculate 4 + 3 + 2 + 1, which equals 10.

```
def sum_up_to(n):
    # Base case: if n is 0, the sum up to 0
is just 0
    if n == 0:
        return 0
    # Recursive case: n plus the sum of
numbers up to n-1
    else:
        return n + sum_up_to(n-1)

# Example usage
print(sum_up_to(4))  # Output: 10
```

This function works by taking the integer we supply (as 'n') and adding to it all the numbers that are 1 below it. It does this by adding n to the sum of n-1 (by calling sum_up_to(n-1) for all the numbers until it hits 0.

Then it works back up the chain to output the final sum. Recursion is one of those magical parts of Python that you will encounter fairly frequently, and it works but it can be quite challenging to visualise in your mind. Please don't worry if you don't quite get it from this simple introduction. There will be time to understand recursion later.

Command line

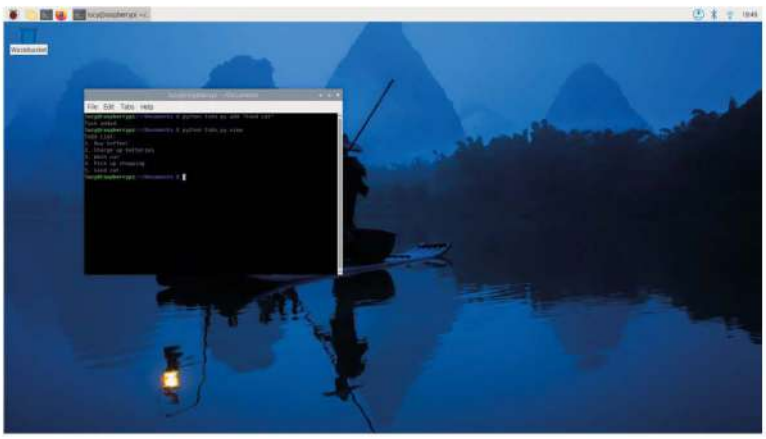
Let's turn what we've learnt about Functions into a useful program. We're going to expand on the ToDo list from the previous exercise. This time we're going to turn it into a more detailed program that runs from the command line.

Enter the code from **todo.py** in Thonny and save it to your disk. Open a Terminal window, navigate to the same directory as your **todo.py** program, and run the program to add, view, and delete tasks.

- To add a task, use: `python todo.py add "Task details here"`
- To view all tasks, use: `python todo.py view`
- To delete a task by its number (as shown in the view), use: `python todo.py delete 1` (where 1 is the task number to delete)

This application demonstrates basic file handling, argument parsing, and simple CRUD (Create, Read, Update, Delete) operations in a

Running our todo.py Python program from the Terminal



Top Tip

What's a tuple

A tuple is a special type of data structure that is similar to a list, only it is immutable (fixed and unchangeable). Once created it can't be changed, only used or deleted.

> We removed the .py extension when moving the file. This is so we can run the command from anywhere in Terminal

Top Tip

Run from Thonny

You can run **todo.py** from Thonny with arguments using the `%Run` command in Shell, for example, `%Run todo.py view`.

command line application using Python. Feel free to extend it with more features, like editing tasks or categorising them.

Let's script it

To turn `todo.py` into a fully-fledged application (or script) that you can run from the command line, follow these steps.

Add a "shebang" line (`#!`) to the start of the program. This tells the command line which version of Python to use. Make sure this is the very first line.

```
#!/usr/bin/env python3
```

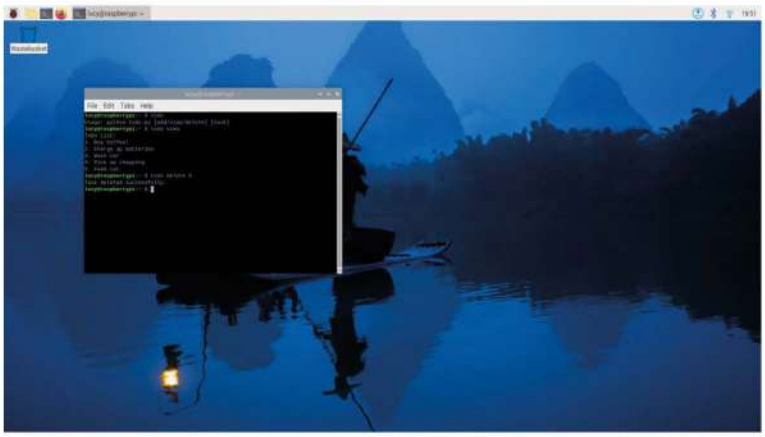
Now make the script executable. Open a terminal window in the same directory as your **todo_list.py** file and enter the following:

```
cd /path/to/todo.py
chmod +x ./todo.py
```

Replace the **/path/to/todo.py** with your own directory. In our case: **/home/lucy/Documents/todo.py**.

The `chmod +x` command adds executable rights to the file, allowing the code to be run from the command line.

▼ Our final ToDo program running directly as a command anywhere in Terminal



Now copy the file from your working directory to a directory on your PATH. The path is a list of locations that terminal checks for files when you run commands in the shell. You can see the current PATH locations with:

```
echo $PATH
```

The directories in the PATH are listed with a colon ":" separating them. A common place to move files to is **/usr/local/bin** (bin stands for "binary"). We're going to copy our **todo.py** file here (you will need to use `sudo` to gain permission):

```
sudo cp ./todo.py /usr/local/bin/todo
```

Notice that we removed the .py extension when moving the file. This is so we can just run the command from anywhere in Terminal using "todo" as we would any other command like ls, cp, or cat. Now either close Terminal and open another window, or enter the following command to refresh the shell.

```
source ~/.bashrc
```

Now you can run the program from anywhere in Terminal using just the command `todo` followed by an option:

- todo add
- todo view
- todo delete

Congratulations now you have a working ToDo program that saves your ToDo list to a file called **todo_list.txt** in your home directory. You can view ToDo's using the todo view option or using `cat ./todo_list.txt`.

There's plenty more you can do with your ToDo program. Above all we hope you've enjoyed learning about Functions and building a fully functional program that adds to your Raspberry Pi Terminal.

todo.py

DOWNLOAD THE FULL CODE:
magpi.cc/github

> Language: **Python**

```
001.  #!/usr/bin/env python3
002.
003.  import sys # Import the sys module to access
      command-line arguments
004.  import os # Import the os module to check if a
      file exists
005.
006.  # Define the path to the file
007.  home_dir = os.path.expanduser('~')  # Gets the
      user's home directory
008.  todo_file = os.path.join(home_dir,
      'todo_list.txt')  # Builds the full path
009.
010.  def load_tasks():
011.      # Load tasks from the file
012.      if os.path.exists(todo_file):
013.          with open(todo_file, 'r') as file:
014.              tasks = file.readlines()
015.              tasks = [task.strip() for task in
      tasks]  # Remove newline characters
016.      else:
017.          tasks = [] # Return an empty list if
      the file does not exist
018.      return tasks
019.
020.  def save_tasks(tasks):
021.      #Save tasks to the file
022.      with open(todo_file, 'w') as file:
023.          for task in tasks:
024.              file.write(f"{task}\n")
025.
026.  def add_task(task):
027.      # Add a new task
028.      tasks = load_tasks()
029.      tasks.append(task)
030.      save_tasks(tasks)
031.
032.  def view_tasks():
033.      # Display all tasks
034.      tasks = load_tasks()
035.      if tasks:
036.          print("ToDo List:")
037.          for idx, task in enumerate(tasks,
      start=1):
038.              print(f"{idx}. {task}")
039.      else:
040.          print("Your ToDo list is empty!")
041.
042.  def delete_task(task_number):
043.      # Delete a task by its number
044.      tasks = load_tasks()
045.      if 0 < task_number <= len(tasks):
046.          del tasks[task_number - 1]
047.          save_tasks(tasks)
048.          print("Task deleted successfully.")
049.      else:
050.          print("Invalid task number.")
051.
052.  def main():
053.      # Check if the user has provided a command
054.      if len(sys.argv) < 2: # The first argument
      is the script name
055.          print("Usage: python todo.py [add/view/
      delete] [task]")
056.          return
057.
058.      command = sys.argv[1].lower()
059.      if command == 'add':
060.          task = ' '.join(sys.argv[2:])
061.          add_task(task)
062.          print("Task added.")
063.      elif command == 'view':
064.          view_tasks()
065.      elif command == 'delete':
066.          if len(sys.argv) == 3:
067.              task_number = int(sys.argv[2])
068.              delete_task(task_number)
069.          else:
070.              print("Usage: python todo.py delete
      [task number]")
071.      else:
072.          print("Invalid command. Use add, view,
      or delete.")
073.
074.  if __name__ == "__main__":
075.      main()
```

Part 01

CDP Studio: Control a robot arm

Use CDP Studio and its Kinematics framework to control a Raspberry Pi-based robot arm

MAKER

Phil King

Long-time contributor to *The MagPi*, Phil is a freelance writer and editor with a focus on technology.

@philkingeditor

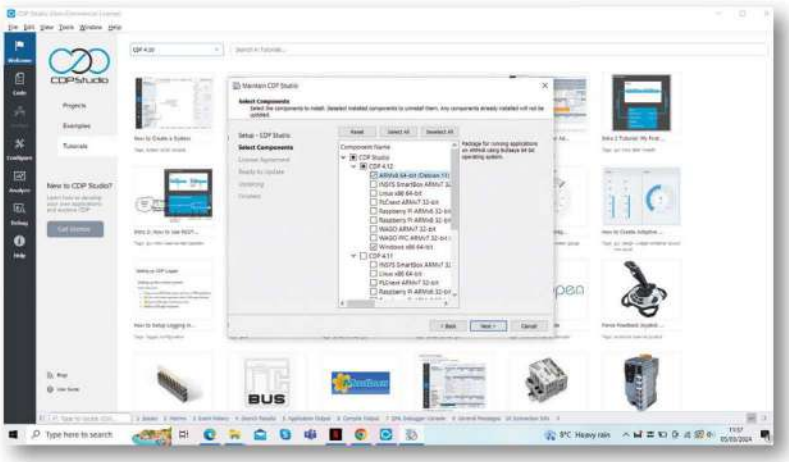
CDP Studio is an 'out of the box' software development tool used by many companies to build industrial control, automation, and edge systems. Yet it's fairly easy to get to grips with its low-code programming environment.

In The MagPi issues 116 and 117, we showed you how to deploy a couple of projects to a Raspberry Pi. This time, we'll be using CDP Studio and its Kinematics framework to program a Raspberry Pi-powered robot arm, the myCobot 280 Pi that we reviewed in issue 137 (**magpi.cc/137**), to perform a pick and place routine. If you don't have the robot arm, you can still run the project and record movement steps to see how they affect the position of a virtual arm shown on screen.

01 Install the software

On your PC, visit **cdpstudio.com/getstarted** and download the free non-commercial version for Windows or Linux. During installation, select the 'ARMv8 64-bit (Debian 11)' component under CDP Studio 4.12, along with the one already ticked for your host PC. You will then be able to deploy projects to the myCobot 280 Pi arm, which uses a 64-bit version of Ubuntu.

If you already have CDP Studio installed, make sure it's updated to version 4.12, then go to Help > Package Manager and select 'Add or remove CDP versions' to add the ARMv8 64-bit (Debian 11) component.



▼ Adding the ARMv8 64-bit (Debian 11) toolkit required to control the myCobot robot arm

02 Download the project

This is a complex project that would be time-consuming to build from scratch, so we'll download it from CDP Studio's GitHub repo. Go to **magpi.cc/recordnplay**, click the green Code button, and select Download ZIP. Unzip the file on your PC. Move the resulting **myCobotRecordNPlay-main** folder to the **CDPStudioWorkspace/systems** folder.

03 Download and build library

To deploy the project to the robot arm, you'll also need the myCobotLib library. Go to the GitHub repo at **magpi.cc/mycobotlib**, click the Code button and Download ZIP. Extract it and then move the resulting **myCobotLib-main** folder to the **CDPStudioWorkspace/libraries** folder. Open the **myCobotLib** project file (with the .pro suffix) in CDPStudio, then right-click its name in the left panel and select Build.

04 Open the project

Now open the **RecordNPlay** CDP project file (.pro) in CDP Studio. If you click the arrow

next to it in the left panel, you'll note that it comprises two main applications. RecordNPlayUI runs the database logic for recording arm movement steps and shows a GUI on the PC to make programming the arm easier. It also has an ArmVisualizer pane that can be used to view the arm positions in 3D. This can be used even if you don't have a real arm connected, so you can still run the project and see how recorded steps affect its movements.

The RecordNPlayIO application is the part of the project that's deployed to the myCobot Pi arm over the network, once paired, enabling CDP Studio to communicate with it.

05 Prepare myCobot

The myCobot arm's Ubuntu OS has a non-standard version of the OpenSSH server. So you'll need to make a small change to a config file so CDP Studio can communicate with it over the network. SSH into the myCobot with the username 'er' at its IP address; the default password is 'Elephant'. Then enter:

```
sudo nano /etc/ssh/sshd_config
```

Locate the line that sets the PubkeyAuthentication parameter and set it to yes (and make sure the line is not commented out). Press **CTRL+X**, then **Y** to exit and save. Then restart the OpenSSH server with:

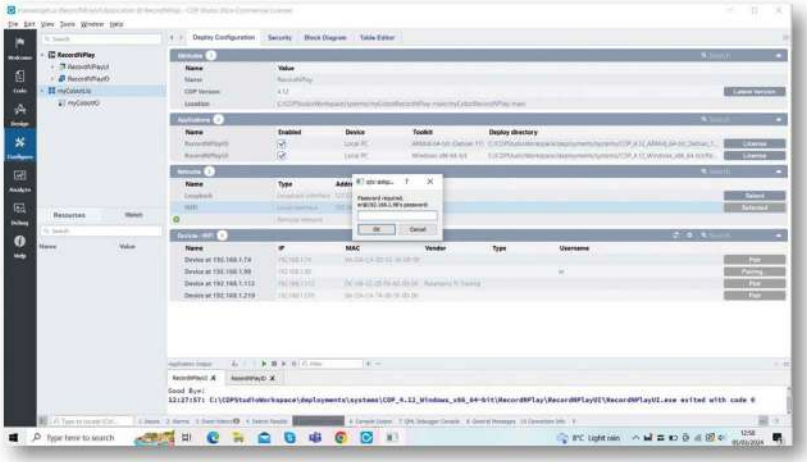
```
sudo systemctl restart sshd
```

06 Pair the arm

Open the Deploy Configuration tab. Under Networks, press the Select button for 'WiFi'. The 'Devices – WiFi' table below should start showing any devices available to pair with CDP Studio.

You'll Need

- Windows or Linux PC
- CDP Studio 4.12 **cdpstudio.com/getstarted**
- myCobot 280 Pi robot arm **magpi.cc/mycobot280**
- myCobot Adaptive Gripper **magpi.cc/mycobotgripper**



▲ Pairing the myCobot robot arm with CDP Studio over the Wi-Fi network

Click the Username field for your myCobot (based on its IP address) and enter 'er', then click the Pair button next to it. You will be prompted to enter the password – the default is 'Elephant'.

Under Applications, change the Device for RecordNPlayIO application to your myCobot device name, then change the Toolkit to ARMv8 64-bit (Debian 11). When you run the RecordNPlay project, this will then be deployed over the network to the robot arm.

07 Run the project

Right-click RecordNPlay in the left panel and select Run & Connect. After a few moments, a new Arm Record'n'Play window should appear, showing the GUI for recording arm movements. First, enter a name for the sequence and click Add. Then click Record to start recording steps. You can move the sliders, but it's a lot easier to move the robot arm around and then click its LED panel button to add each position as a step. Recorded steps are shown in a list and can be updated or deleted individually using the buttons at the bottom right. The 'Step duration' bar sets the time for which an arm position is held.

Try recording some steps and then hit the top-left Play button to play the sequence. If you have a myCobot arm connected, it should follow the movements you recorded; if not, select ArmVisualizer in the project's left panel, then the DHChain Visualizer tab to view a 3D representation of the arm with its six joints. As you move between two steps, the visualisation shows both and the movement of the arm's head with a red line. The gripper status is indicated by a green (closed) or grey (open) dot.

08 Pick and place

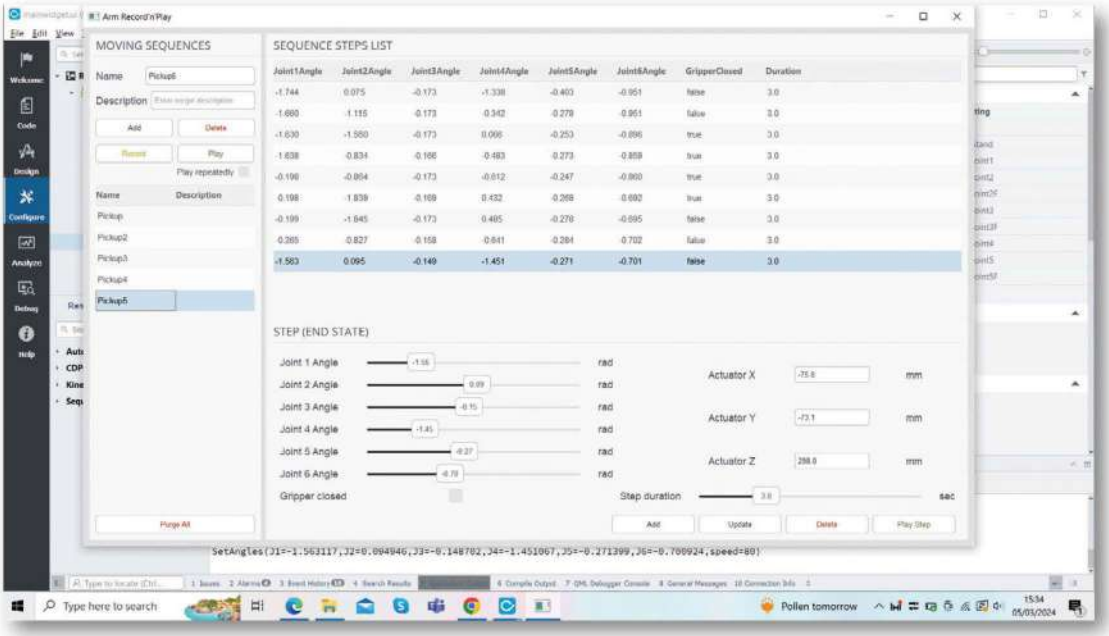
Now, let's get our arm to pick up an object and place it down in another location. We placed a pencil sharpener on top of a biscuit tin, high enough to give the arm plenty of space to pick it up without the bottom of the gripper hitting the table.

> "We placed a sharpener on top of a biscuit tin"

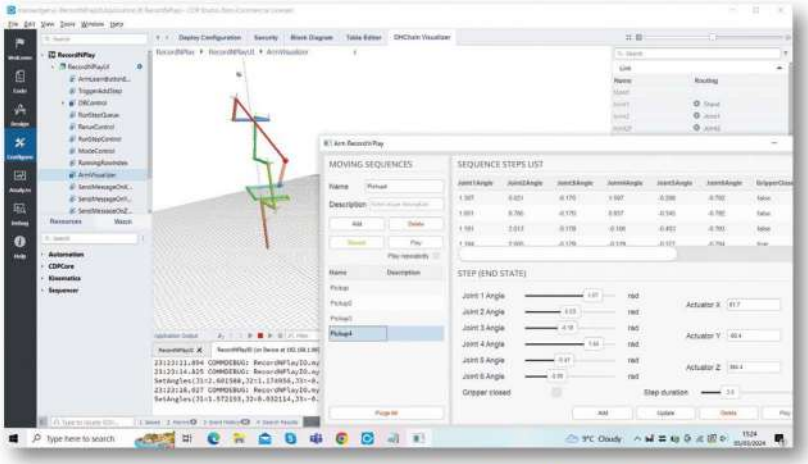
Top Tip

Another arm

While this project is designed for the myCobot 280 Pi, you could use a different robot arm – you'd just need to create a new IO library to communicate with it.



▶ Sequences of steps are recorded in the GUI, so you can play them back to make the robot arm follow them



As you play a sequence, you can see the effects on a virtual arm in the ArmVisualizer pane

The myCobot's adaptive gripper attachment can be used to pick up and drop objects

Move the arm between positions and press the LED panel button to record each step. You can also open and then close the gripper manually to program it. Make sure the arm is stationary, in the right position, before closing it. Then lift the arm up and move it round and down to where you want to place the object. After opening the gripper to drop it, move the arm straight up so you don't bump into the item. You can adjust step positions in the GUI if needed. The steps are stored in an SQLite database too, so you could always edit that manually.

Top Tip

Sturdy base

You'll need a base for your robot arm, to stop it falling over as it moves. We used the G-Base 2.0 to clamp ours to a table.

09 Kinematics

This project makes used of CDP Studio's Kinematics framework, in the form of the DHChain component. The basic concept of kinematics is that if you input joint angles for a robot arm, or chain of links, the framework can calculate the end position in 3D space – as shown in our project's ArmVisualizer pane, with the X/Y/Z coordinates shown in the Arm Record'n'Play GUI.

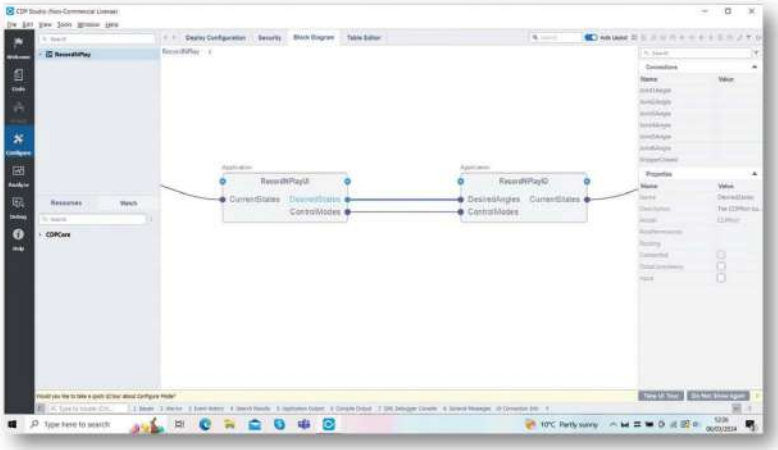
The method can also be used in the reverse direction, to convert a desired 3D end position into the required joint angles; this is known as 'inverse kinematics'.

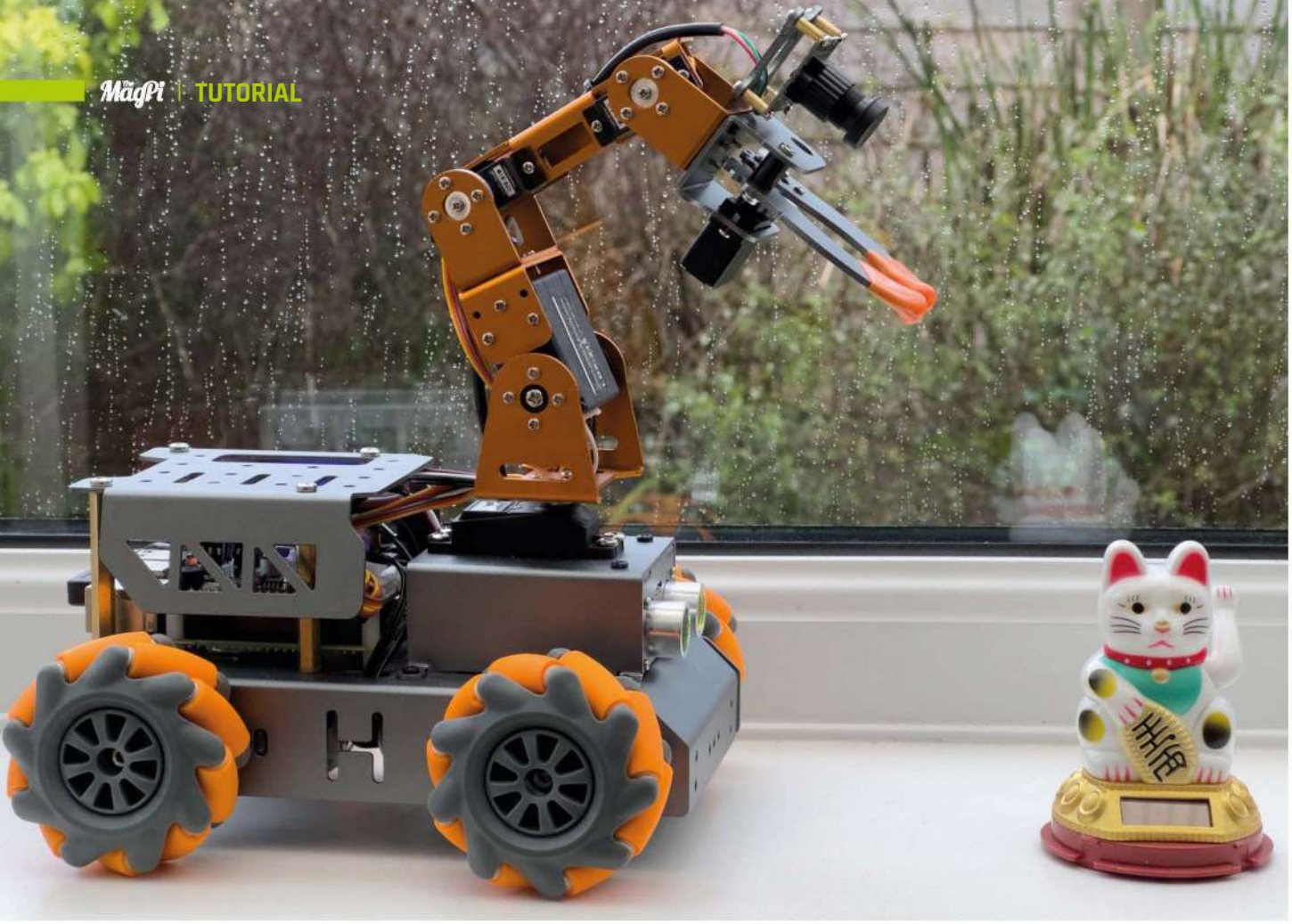
Kinematics has many uses in the field of engineering, helping to calculate positions and velocities of moving parts such as those in an industrial robotic arm, or a bionic limb or exoskeleton. An example real-world case is the use of CDP Studio and kinematics is for controlling deck cranes on ships.

10 Exploring the project

You can click the Block Diagram tab to see how the project's block-based components have been put together. At the highest level, there are two main blocks: for the RecordNPlayUI application for the GUI step recorder, and the RecordNPlayIO one for communicating with the myCobot arm. Here, the UI block's DesiredStates port links to the IO block's DesiredAngles port; it sends the angles set for the six joints, along with the gripper status, so that the arm will move accordingly. The ControlModes link is used to determine whether the arm should maintain a position or be allowed to move freely, for when you're recording moves. CurrentStates is a feedback port from IO that's used by UI to know what is the current position of the arm joints, gripper and LED button; this information is used by the recording process in UI.

There are two main application blocks, the UI step recorder running on the PC, and the IO for communication with the arm





Programming an electronic brain

How the Robot Operating System helps you build complex robots

Figure 1
This robot is called the MasterPi and is made by Hiwonder: hiwonder.com

Figure 2
The USB connection on the right is for the camera on the robot arm

Rob Miles

Rob has been playing with hardware and software since almost before there was hardware and software. You can find out more about his so-called life at **robmiles.com**.

Getting into robot development has never been easier or cheaper. You can pick up a robot kit for not much more than the price of a video game. If you spend a little more money, you can get one with a robot arm on top, like the one shown in **Figure 1** above. Many of the robots use mecanum wheels, which allow the robot to move in any direction.

Figure 2 shows what is controlling the robot. A Raspberry Pi sits underneath a 'HAT' which manages the power supply (two 18650 lithium batteries) and the signals to control the motors and the robot arm. The robot runs a set of Python programs which control the hardware and allow it to perform preprogrammed tasks.

Figure 3 shows the remote-control application you can use to tell the robot what to do. The robot hosts a Wi-Fi access point to which you connect the mobile application. You can then select from pre-built behaviours. This works well, but what if you want to do more? The author was very keen to use his robot as a platform for learning the Robot Operating System (ROS). So that is what he has decided to do. The robot is presently driven by Python programs; the idea is to turn these into ROS nodes. But first, we must learn a bit about ROS itself.

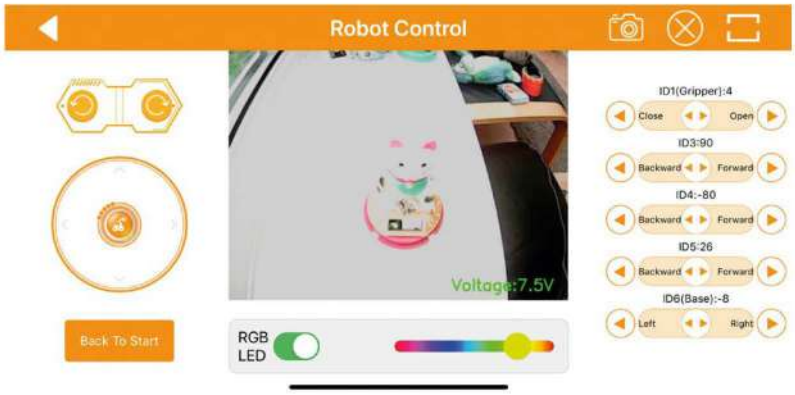


Figure 3
This application is running on an iPhone. A version of the code is also available for Android

COMING UP ROS-ES

An operating system is something you add to hardware to make it useful. Examples are Windows, MacOS, or Linux. The operating system takes the raw ability of the hardware (running programs, reading keyboards, saving data, displaying images on screens, etc.) and provides a user interface. The operating system lets you select the program you want to run. When you start the program, the operating system fetches the selected program from mass storage and then performs the instructions in the program. Some of the instructions will ask the operating system to do things; for example, a word processor will ask for a document file to be loaded into memory.

ROS takes the abilities of a computer system and makes it useful to a robot creator. ROS lets you break a system down into cooperating components. Components are created inside packages, which makes it easy to manage complex solutions. There are many pre-built components that you can incorporate into your solutions. ROS also provides tools you can use to express the physical design of your robot (or other mechanical system controlled by ROS elements) and simulate behaviour in a virtual (i.e. computer-generated) environment. ROS is a rich and complex system which can take a while to master. It provides effective solutions to robotic problems that we aren't even aware we have. Learning it will hurt your head a bit, but it is worth the effort.

INSTALLATION

ROS sits on top of the computer operating system and is closely coupled to it. This means that the versions of ROS you can use are determined by the operating system on your computer. You can run ROS directly on a Windows PC, but the installation is not for the faint-hearted as it involves compiling the ROS program source. It is much easier to use a Linux-based machine for which you can obtain compiled binary versions of ROS. On a Windows PC, you can use the Windows Subsystem for Linux (WSL) to make a Linux environment into which you can install ROS. There is a step-by-step guide to the installation process in the GitHub repository for this article (**hsmag.cc/RosIntro**). The process uses the amazing Docker tool, which makes it possible to host any version of Linux on your machine, whatever its architecture or operating system.

ROS is a large application that needs plenty of memory to run. It will not fit on smaller devices as it needs at least 4GB of RAM. If you are using a Raspberry Pi, you can add a swap file to your system which uses file space to extend main memory. However, if you do this, you may find that your SD card (the place where your files are stored) wears out as the operating system will continually write to the swap file as programs are started and stopped. If you are feeling brave (and have a few spare SD cards), there are instructions on how to do this in the installation guide.

INSIDE THE WORLD OF ROS

ROS breaks a robot application down into nodes. A node is implemented as a running program. The nodes talk to each other in a well-defined way, and they cooperate to keep the robot running. The nodes can all run as individual tasks on a single computer, or they can run on lots of different processors connected to the local network.

Figure 4, overleaf, shows a ROS installation which contains a controller, a robot, and a camera. Perhaps we want to make a robot litter picker which will wander around searching for litter. The robot is constructed as three devices: a controller, a robot rover, and a camera. Each device is running one or more nodes that form part of the ROS solution.

Nodes provide services to other nodes and accept commands from them. In addition, any node can publish data items that any other node can subscribe to. There are lots of advantages to organising a robot application this way. It is easy to move nodes between devices. The connections between the →

YOU'LL NEED

- **A reasonably powerful desktop computer, laptop, or Raspberry Pi (preferably a 4 or 5 with 4GB of RAM) to run the development environment and ROS**
- **Your own little robot or the parts to make one**

QUICK TIP

You can use your knowledge of ROS to power a pre-built robot or to take control of a robot that you have created yourself out of components you have chosen. There are tips on hardware choices in the GitHub repository for this article: **hsmag.cc/RosIntro**

nodes are well-defined. We could move all the nodes onto a single powerful computer, or replace the vision system with one that uses a different camera.

We don't have to write the code for all the nodes. ROS also includes a library mechanism that makes it easy to import node code. To properly understand all of ROS, you must understand all the problems that it solves, and there are many of those. Let's start with something simple. How do ROS nodes help us get our 'litter picker' robot going?

QUICK TIP

ROS is not just good for robots. If you've got a complex application that you want to break down into cooperating components, you should look at what ROS offers.

CLEANING UP

To activate the litter-picking robot, the user could press the 'Search' button on the controller device. The buttons on the front panel of the controller are managed by the **buttons** node, which publishes the button states on a topic called 'buttons'.

The **manager** node has subscribed to this topic, so it receives a message informing it that the Search button has been pressed. The **manager** then sends a service request to the **vision** node asking, "Can you see anything?"

The vision system has subscribed to frames of image data published by the **camera** node and is looking for litter in each frame it receives. If the **manager** gets a response indicating that some litter has been spotted and giving a direction to that litter, the **manager** sends commands to the **motor** node in the robot to head that way. The **manager** also publishes the message 'moving' to a robot status topic.

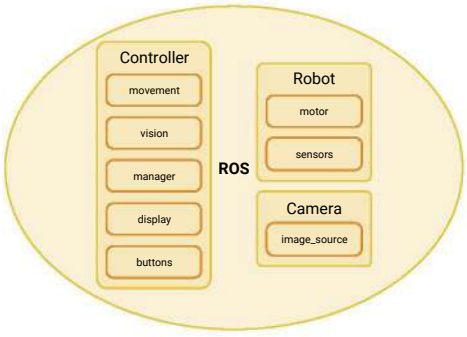


Figure 4
There are eight nodes in this system

The **display** node has subscribed to this topic, so the status 'moving' is displayed. As the robot moves, the **sensors** node on the robot would be publishing information about things the robot is detecting around it. ROS provides the environment in which all this would be made to work. We design the structure of the published data, write the code for each node, and then decide how the nodes will interact, but ROS sits underneath and makes everything work.

PUSH THE BUTTON

We can discover how ROS does this by considering the very start of this process: the front panel of the robot. At the very least, the robot will have a button to make it start and a button to make it stop. It should also have a way of indicating what it is doing, perhaps a coloured light or a text display. The diagram in **Figure 4** shows that the Controller is running a **buttons** node which receives user commands and a **display** node that displays the robot status.

Let's look at the code in this node, which is implemented by a Python class called **ButtonPublisher**. This class extends the **Node** class which is part of ROS. An instance of **ButtonPublisher** class is created when the robot system starts up. Below, you can see the constructor method which runs when a **ButtonPublisher** is created.

```
class ButtonPublisher(Node):

    def __init__(self):
        super().__init__('button_publisher')
        self.buttonReader = ButtonReader()
        self.start_publisher()
        self.start_timer()
```

The constructor first calls the **__init__** method for the **Node** parent class, supplying the call with the name of the publisher. This tells ROS that there is a new node called **button_publisher** in town. The next statement in **__init__** creates a **ButtonReader** object which will be used to read the buttons on the physical console.

The **ButtonReader** reads the buttons from the controller hardware, perhaps by using GPIO pins (although it could also connect to a computer keyboard or touchscreen). Next, the constructor calls **start_publisher** to create a publisher to publish button events to anyone who is subscribed to them. Then it calls the **start_timer** method to start the button scanning timer. There is not much code in the **start_publisher** method:

```
def start_publisher(self):
    self.publisher_ = self.create_
publisher(String, 'buttons', 10)
```

The publisher to be used by **ButtonPublisher** is stored in a class member called **publisher**. The **create_publisher** method is inherited by

ButtonPublisher from its **Node** parent class. The **publisher** method accepts three parameters. The first indicates that the publisher will publish a String, the second gives the topic for the published message (in this case, 'buttons'), and the third parameter (the value 10) means 'keep the last ten published items and use the "RELIABLE" level of quality of service' – which means that published items are guaranteed to arrive at the receiver. The second method called by the **ButtonPublisher** constructor is **start_timer**:

```
def start_timer(self):
    timer_period = 0.1  # seconds
    self.timer = self.create_timer(timer_period,
self.timer_callback)
```

This method creates a timer which fires ten times a second. Each time the timer fires, the **timer_callback** method is called.

```
def timer_callback(self):

    button = self.buttonReader.scan_buttons()
    if button!="":
        msg = String()
        msg.data = button
        self.publisher_.publish(msg)
        self.get_logger().info('Publishing: "%s"'
% msg.data)
```

The **timer_callback** method calls the method **scan_buttons** on the button reader. This scans the buttons and returns the name of the button which is pressed, or an empty string if no buttons are pressed. If a non-empty string is returned, the callback publishes the name of the button that is pressed on the 'buttons' topic.

The final part of the button node program is the mechanism which starts the node itself. This is performed by the **main** method in the node code which is called when the node is loaded.

```
def main(args=None):
    rclpy.init(args=args)

    button_publisher = ButtonPublisher()

    rclpy.spin(button_publisher)

    # Destroy the node explicitly
    button_publisher.destroy_node()
    rclpy.shutdown()
```

The **main** method creates a **ButtonPublisher** instance and then calls the **button_publisher** method on this. The **rclpy.spin** function keeps the node alive; responding to events and managing callbacks. It ends if the node is terminated by ROS, at which point the publisher node is destroyed and the node shuts down.

All the nodes in the robot are started in this way. The robot controller code can contain a launch method which starts all the nodes when the robot begins running.

SUBSCRIPTION MODEL

We know how a node running in a robot can post a message on a topic. Next, we need to consider how another node can receive it.

```
class DisplaySubscriber(Node):

    def __init__(self):
        super().__init__('display_subscriber')
        self.subscription = self.create_
subscription(
            String,
            'buttons',
            self.listener_callback,
            10)
        self.subscription  # prevent unused
variable warning

    def listener_callback(self, msg):
        self.get_logger().info('Button: "%s" was
pressed' % msg.data)
```

The code above defines a **DisplaySubscriber** class which subscribes to the **buttons** topic and calls the **listener_callback** function each time a message is received. This listener callback simply logs the button name (although it could put it on a text display). Code in the **manager** node could also subscribe to the **buttons** topic so that the manager is informed when a button is pressed.

PACKAGE HOLIDAY

Code for ROS applications is organised into packages. A package brings together a set of related behaviours. We could create a package called **front_panel** which contains the code for the **buttons** and **display** nodes.

Figure 5, overleaf, shows the package files for the **front_panel** package. The two highlighted files contain the code for the **buttons** and **display** Python nodes that we have seen above. The three files at the bottom of the package are what ties the package together. The **package.xml** file contains a description of the package and identifies any dependencies that the package has.

QUICK TIP

The most recent version of ROS is called ROS2. This is the one you should be using.

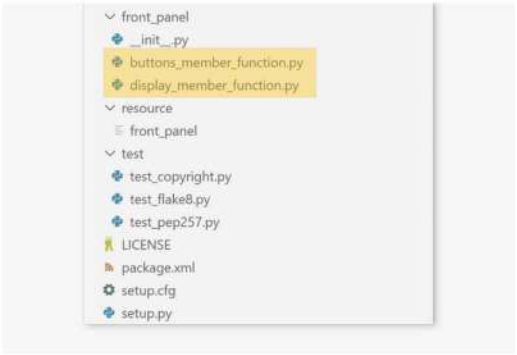


Figure 5 ROS can generate a template package for you to fill in

One package can use elements from another, and this is where you would identify the source packages. The **setup.cfg** file specifies where the files are to be placed, and the **setup.py** is a chunk of Python which is run to set up the project. This file is worth looking at:

```
from setuptools import find_packages, setup

package_name = 'front_panel'

setup(
    name=package_name,
    version='0.0.0',
    packages=find_packages(exclude=['test']),
    data_files=[
        ('share/ament_index/resource_index/
packages',
            ['resource/' + package_name]),
        ('share/' + package_name, ['package.
xml']),
    ],
    install_requires=['setuptools'],
    zip_safe=True,
    maintainer='rob',
    maintainer_email='rob@hullpixelbot.com',
    description='Provides a front panel containing
buttons and a text display',
    license='Apache-2.0',
    tests_require=['pytest'],
    entry_points={
        'console_scripts': [
            'buttons = front_panel.buttons_member_
function:main',
            'display = front_panel.display_member_
function:main'
        ],
    },
)
```

Above is the **setup.py** file for the `front_panel` package. The most important part of this file is the entry point list at the bottom. It specifies the Python files that are to be run when the nodes are activated. If we added a third node to the panel (perhaps a buzzer controller), we would create the Python code for the node and then add the entry point here. The above package serves as a template for any future packages you want to create. Just add the Python code into the same folder as the existing nodes and configure **setup.py** with the new entry points. Your code can then be run as a node in the robot system.

BUILDING THE PACKAGE

Now that we have our package, the next thing we need to do is build it. This is the process of collecting all the package components and getting them into a state where ROS can run them. Python programs are not compiled, but ROS nodes can also be created from C++ code, which does need to be compiled before it can run. A package that contains C++ code has a slightly different format and contains a **CMakelists.txt** file which describes how to build the code. It makes creating a package a bit trickier, but it does mean that you can use both languages in your solution. Furthermore, you can import packages containing nodes programmed in C++ and they will work alongside your Python nodes. The ROS system provides a command called `colcon` (collect components) that performs the build process:

```
colcon build --packages-select front_panel
```

The command above would be performed to build the `front_panel` and make the files ready to run. So, at last, we can run our robot nodes. There's just one more thing we need to know about (sorry), and that is all to do with sourcing our commands. We talk to the operating system from within a 'shell' environment which accepts our commands and then performs them. On Linux, this is usually the 'bash shell'. Some of the commands are 'built-in' to the shell; for other commands, the shell will go off and look for a program to run. We can tell the shell where to look for our robot node code:

```
source install/setup.bash
```

The `source` command means 'find this file and execute it as if it has been typed in'. The command file is called `setup.bash`, and it is created for us by `colcon` when the package is built. Once we have performed this command, we can use the `ros2 run` command to start our two nodes running:

Figure 6 shows two bash shells running the `buttons` and `display` nodes. The two nodes were

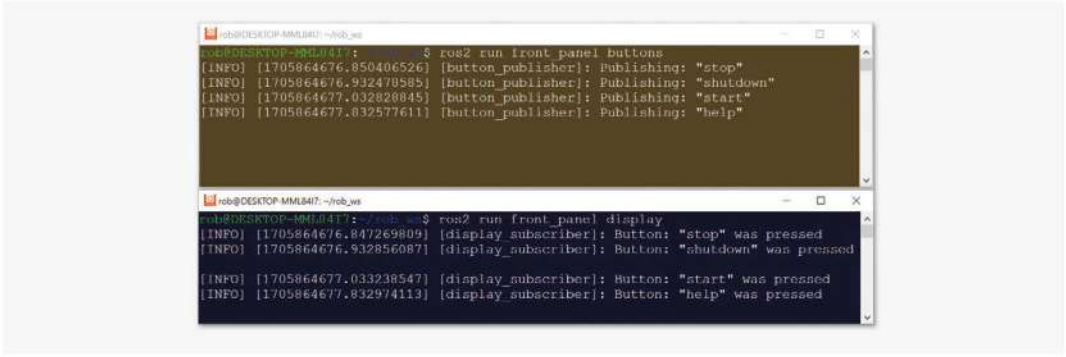


Figure 6
These packages are running on a PC under the Windows Subsystem for Linux, but they could just as easily be running on a Raspberry Pi

started using the commands at the top of each shell. The **buttons** node has published four button messages and the **display** node has displayed them.

ROS COMPLICATIONS

At this point, you might be thinking one of two things: 'Blimey, this is complicated' or alternatively, 'Blimey, this is powerful'. The complexity is there for a reason. We want to be able to break our robot controller down into smaller reusable components which can be easily swapped without affecting the rest of the application.

One robot might use a physical button to start it, but the next might have a touchscreen. Because this input is managed by a node, we just need to swap out the package node for another. Later, we might decide to combine the camera and the controller into a single device. In that situation, we just have to change the **image_source** node to run on the controller, and everything which uses images will just work.

The package folder contains everything needed to build and run a particular set of robot nodes. There are many pre-built packages for robot interfaces and behaviours that you can slot into your robot and interact with via their services and topics.

NODE MANAGEMENT

One of the many wonderful things about ROS is the way that things 'just work'. If you want two nodes to communicate, you just run them and they can magically see each other.

ROS works over your local area network (LAN). Network communications are based on the DDS (Data Distribution Service) standard, which provides the discovery protocol by which nodes can identify themselves and find other nodes. ROS provides commands we can use to discover nodes, topics, and services.

Figure 7 shows the output from three ROS commands which can be used to view an active ROS installation. The **node list** command lists all the active nodes – in this case, the **button** and the **display**. The **topic list** command shows all the active topics. The **buttons** topic is active, along with two which are provided by ROS. The final command shows all the services that are provided. Note that the button and the display components expose a set of services which can be used to work with them. This makes it possible for a ROS application to automatically discover what nodes can do and how to use them, which makes possible self-configuring systems. We can see this in action if we start up the rqt tool, which is supplied as part of ROS. This contains a range of plug-ins you can use to create interfaces with your ROS application.

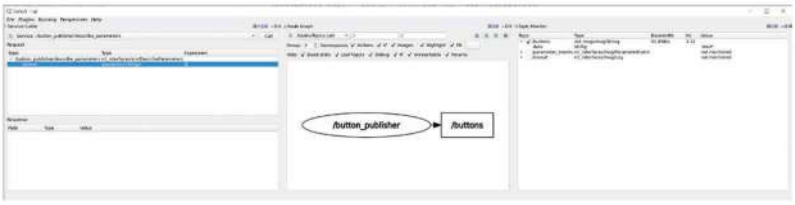


Figure 8
You can add plug-ins and lay out the display as you like

The rqt window in **Figure 8** shows a view of the topics available on the left, a diagram showing the active robot topics in the middle, and a log of button topic messages on the right. We can see that the last message that was sent was the 'start' message. These views are all updated in real time.

Now we know how ROS applications are structured and how the components interact. In the next article, we'll build on this.

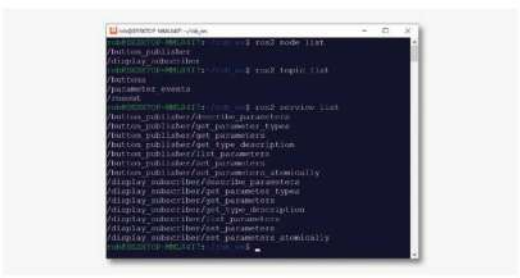


Figure 7
If other ROS nodes were active, they would appear in this output, too


FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE
START
WINNER
NEW 2024 UPDATE
RETRO GAMING WITH RASPBERRY PI
3RD EDITION
180 PAGES OF VIDEO GAME PROJECTS
BUILD AN ARCADE MACHINE
PLAY & CODE GAMES!
File manager
Change color
Change icon
Extra blocks
Save dump
Load dump
Hexeditor
Defragment
Reformat
/ : Select
: OK
: Cancel
Display the list of the files found in the selected VMU.

Raspberry Pi 5 Cases

Group Test

Phil King tests some of the best cases available for Raspberry Pi 5

Raspberry Pi 5 is one cool single-board computer, but it can look even cooler inside a smart case. More importantly, a case can also help it to keep the board cool under a heavy workload, by using thermal pads, a heatsink, fan, or a combination of all three. To test the cooling effectiveness, we performed a stress test using all four of the CPU cores at the default maximum of 2400GHz. We also checked the Wi-Fi signal to see if the case interfered with it. In addition, we assessed each case's capacity – for mounting a HAT etc – and the access to Raspberry Pi 5's ports. It's a best case scenario!

Official Raspberry Pi 5 Case

Raspberry Pi | magpi.cc/case
£10 / $10

Coming in red/white or black, the official case comprises three plastic sections that snap-fit together, so assembly is a cinch. The middle section includes a clear insert with a fan that connects to Raspberry Pi 5's Fan port. A small SoC heatsink is also included.

Capacity: By removing the clear insert and fan from the middle section, you can mount a HAT. Or you could use a booster header to lift it above the fan.

Port access: All side ports are accessible via cutaways, along with the power button. The middle section features a wide slot to enable access to the GPIO pins when the lid is removed; it could also be used for routing cables for a camera or other accessories.

Cooling: The variable-speed fan's cooling is aided by vents in the bottom of the case, plus the SoC heatsink. It works even with the lid on, due to a circular gap. In our tests, it prevented throttling even under a heavy workload, but some other cases performed better.

Wi-Fi signal: As you'd expect, the plastic case has very little effect on the signal, which remained strong.

Verdict
A well-thought-out, versatile design that covers most uses.

ICE Tower CPU Cooler

The Pi Hut | magpi.cc/icetower
£16 / $17

Its cool-looking tower design is enhanced by RGB LEDs making it glow in various colours. Is it really a case? Well, it does have a plastic base, connected via screws and bolts to two metal mounting brackets and the main fan/heatsink section on top. Assembly is a little fiddly (especially the brackets), but not too tricky. The heatsink it connected to the SoC via a thermal pad.

Capacity: You won't be mounting any HATs with that large fan and heatsink sitting atop Raspberry Pi!

Port access: The open design means access to all the ports and GPIO pins is unhindered.

Cooling: The vertically mounted fan blows air onto the aluminium fins of the large heatsink, whose base is connected to Raspberry Pi 5's SoC via a thermal pad. This results in excellent cooling performance. By default, the fan doesn't even kick in until the temperature reaches 60°C (which didn't happen during our tests), but it could prove more useful if you're overclocking Raspberry Pi 5.

Wi-Fi signal: The presence of that metal heatsink does have an effect on the signal, but not too much.

Verdict
With its unusual design and lighting, it looks very cool… and delivers great cooling.

Argon NEO 5

Argon 40 | magpi.cc/argonneo5
£18 / $19

A three-piece design – with two aluminium sections and a plastic base – the NEO 5 oozes style and quality with its red/black colour scheme and solid feel. The middle section is especially impressive, with a fan next to curved fins to aid cooling. Assembly is aided by a guide in a small booklet.

Capacity: With the top part removed, you could mount a HAT with a booster header to lift it over the fan. A special NVMe version of the case is also available.

Port access: The middle section has cutouts for the main ports, camera/display, PCIe, UART, RTC, and the GPIO pins (with a helpful labelled strip on the side).

Cooling: The fan's effect is aided by vents in the middle section and base, plus a couple of thermal pads for the SoC and PMIC. This results in very good cooling performance.

Wi-Fi signal: The mainly metal case does result in a noticeable reduction in signal strength and quality.

Verdict
A stylish, quality case with great cooling performance.

KKSB HAT Case

The Pi Hut | magpi.cc/kksbhat
£16 / $16

The tallest case in the group, it's 56mm high, and is designed to accommodate a Raspberry Pi 5 with a HAT mounted on top. The anodised aluminium case has no built-in cooling, but you could add an Active Cooler or heatsink. There's plenty of room, and a GPIO booster header is supplied if needed to lift a HAT up slightly. Assembly is tricky, as you need to remove both side panels and slide in Raspberry Pi, screw it in place with tiny screws, and then add a HAT afterwards.

Capacity: The high headroom means there's plenty of room to mount HATs on top of Raspberry Pi. Getting things in and out of the case and securing them in place is fiddly, though. We'd have preferred a removable lid.

Port access: The main ports are accessible via cutouts. There are slots for camera connections and PCIe ribbon cables. You'll need to remove one or both side panels to access the other ports.

Cooling: There's no built-in cooling, but the KKSB can be used with an Active Cooler, or most other coolers and heatsinks. Lots of slots in the case aid ventilation.

Wi-Fi signal: Maybe it's all those ventilation slots, but the signal is hardly affected.

Verdict
Not the most user-friendly design and no built-in cooling, but good capacity.

Passive Cooling Open CNC Case

EDATEC | magpi.cc/opencnccase
£7 / $8

Raspberry Pi 5 is sandwiched between the two case sections, with no side pieces. Each grooved aluminium piece is fitted with several thermal pads to aid passive cooling, including for Raspberry Pi 5's SoC, PMIC, and wireless module. The two case sections are secured with long bolts.

Capacity: With the open design and GPIO header access, you can mount a HAT just above the top section.

Port access: With no side pieces, access to ports is unfettered, with cutouts for the GPIO pins, PoE header, PCIe and camera/display ports, plus UART and RTC battery connectors.

Cooling: Highly effective passive cooling is provided by the numerous thermal pads and grooved aluminium case pieces. It keeps Raspberry Pi 5 cool (37.8°C) even under a heavy workload.

Wi-Fi signal: The metal case does result in a noticeable reduction in signal strength and quality.

Verdict
Great port access and some impressive passive cooling performance.

ABS Fan Case

The Pi Hut | magpi.cc/absfancase
£15 / $16

While it doesn't feel the most solid or weighty case, its two main sections have plentiful vents and you get a choice of coloured stripes to stick on the lid! An Armour Lite V5 fan-equipped heatsink sits on top of Raspberry Pi 5 via five thermal pads.

Capacity: The heatsink takes up a fair amount of space in the case. You could still mount a HAT with a GPIO header booster and the lid off.

Port access: The main side ports are accessible via cutouts. With the lid removed, you can access all the others.

Cooling: Five thermal pads connect the SoC, PMIC, RP1, RAM, and wireless module to the heatsink with built-in fan. Considering this, cooling performance was not quite as good as we expected, but decent.

Wi-Fi signal: There's some effect on the strength and quality, possibly due to the metal heatsink.

Verdict
Middling cooling performance in a lightweight case.

Pibow Coupe 5

Pimoroni | magpi.cc/pibowcoupe5
£10 / $10

Like previous editions of the classic Pibow, this case comprises multiple acrylic layers that need to be assembled to surround Raspberry Pi and then secured with four long bolts. It's like doing a 3D jigsaw puzzle. The top of Raspberry Pi is left open, which will suit some use cases better than others.

Capacity: The case's open top design means you can mount any HAT you like.

Port access: The USB and Ethernet ports sit on top of the case, while cutouts enable access to the rest. GPIO pins are fully accessible.

Cooling: There's no built-in cooling, but the Pibow can be used with an Active Cooler or standard heatsink.

Wi-Fi signal: As you'd expect from an open-top case, the signal strength and quality are very good.

Verdict
A multi-part, open-top case with no built-in cooling, but very versatile.

FLIRC Raspberry Pi 5 Case

The Pi Hut | magpi.cc/flircrpi5
£14 / $15

Like previous models, this smart-looking FLIRC case is designed for use with a media centre setup. The main aluminium section also acts as a heatsink via a thermal pad stuck to Raspberry Pi's SoC. There's also a mini power button.

Capacity: There's no room for a HAT inside the case.

Port access: The regular side ports are all accessible via cutouts, but the internal ones – including PCIe and GPIO pins – are not.

Cooling: The metal case acts as one large heatsink via a thermal pad on the SoC, providing effective passive cooling.

Wi-Fi signal: The metal case does reduce the signal strength and quality somewhat.

Verdict
A smart and robust case for a media centre.

Passive Cooling CNC Case

EDATEC | magpi.cc/cnccase
£15 / $19

As with the Open CNC Case, the two case pieces are aluminium; here they lack grooves, but retain the thermal pads. The bottom part also has sides with cutouts for the ports. The result is a sleek-looking case with a snug fit.

Capacity: The GPIO header cutout enables the mounting of a HAT above the top of the case.

Port Access: Cutouts enable full access to the main side ports, along with GPIO pins, PoE header, PCIe and camera/display ports, and UART and RTC battery connectors.

Cooling: As with the Open version of the case, the numerous thermal pads (for SoC, PMIC, wireless, and underside) provide highly effective passive cooling.

Wi-Fi Signal: Fully enclosing Raspberry Pi 5 in a metal box has an adverse effect on the wireless signal.

Verdict
A metal case with great cooling, but Wi-Fi signal isn't great.

Argon ONE v3

Argon 40 | magpi.cc/argononev3
£25 / $30

The biggest case of the bunch, it features a daughterboard that plugs into the side of Raspberry Pi 5. As well as rerouting video and power to full-size HDMI and USB-C ports, its RP2040 chip handles cooling and power management. There's also an IR receiver to use with a remote (supplied separately). A removable magnetic flap in the case top reveals the GPIO pins and labelled strip. An NVMe version of the case is also available.

Capacity: Despite the size, it's not possible to mount a standard HAT inside the case.

Port Access: USB and Ethernet are accessible, along with power and two full-size HDMI ports.

Cooling: An angled fan in the top case section is aided by two thermal pads. You need to install a script for fan control. Cooling performance wasn't the best, especially under stress.

Wi-Fi Signal: Strength and quality were only slightly lower than with the official case.

Verdict
A feature-packed case that's ideal for a media centre or desktop system.

Cooling comparison

We checked the typical CPU temperature when idling and under stress (with all four cores running at 2400MHz).

Case	Idling (°C)	Stressed (°C)
Official Raspberry Pi 5 Case	47.2	65.9
ICE Tower CPU Cooler	31.2	48.8
Argon NEO 5	28.5	42.2
KKSB HAT Case*	–	–
EDATEC Passive Cooling Open CNC Case	25.7	37.8
ABS Fan Case	45.5	65.3
Pibow Coupe 5*	–	–
FLIRC	33.4	49.9
EDATEC Passive Cooling CNC Case	27.0	40.0
Argon ONE v3	39.5	67.5

* No built-in cooling.

Wi-Fi signal

We tested the average strength and quality of Raspberry Pi 5's Wi-Fi signal for each case.

Case	Strength (dBm)	Quality (%)
Official Raspberry Pi 5 Case	-43	96
ICE Tower CPU Cooler	-59	73
Argon NEO 5	-59	73
KKSB HAT Case	-48	90
EDATEC Passive Cooling Open CNC Case	-59	73
ABS Fan Case	-57	76
Pibow Coupe 5	-46	93
FLIRC	-59	71
EDATEC Passive Cooling CNC Case	-65	64
Argon ONE v3	-50	86

Home Assistant **Yellow**

▶ Home Assistant ▶ **magpi.cc/hayellow** ▶ £107 / $135 (without CM4)

Compute Module-powered home automation is only moments away with this powerful kit. By **Rob Zwetsloot**

SPECS

DIMENSIONS:
123mm x 123mm x 36mm

POWER:
12v/2A barrel DC jack, PoE+ IEEE 802.3at-2009 Class 3 or 4

EXTRA FEATURES:
M.2 SSD slot, stereo audio DAC with 3.5mm jack output, enhanced wireless (Silicon Labs MGM210P Mighty Gecko Module), RTC

▲ The case looks great and houses everything neatly inside

Raspberry Pi for home automation and other forms of IoT is what a lot of people love to use it for, and thanks to that the hardware and add-ons that work with Raspberry Pi are easy to come by. Software-wise as well, one of the more popular systems is Home Assistant, which has been around for nearly as long as Raspberry Pi has itself. You can even find it in Raspberry Pi Imager.

It's no surprise then that its branded hardware is pretty high quality. Powered by a Raspberry Pi Compute Module 4 (which you can supply yourself to keep the price down), and with PoE (power over Ethernet) capabilities, it's a remarkably competent yet simple device on paper, picking up on any IoT-enabled hardware on your local network and very quickly letting you start controlling it.

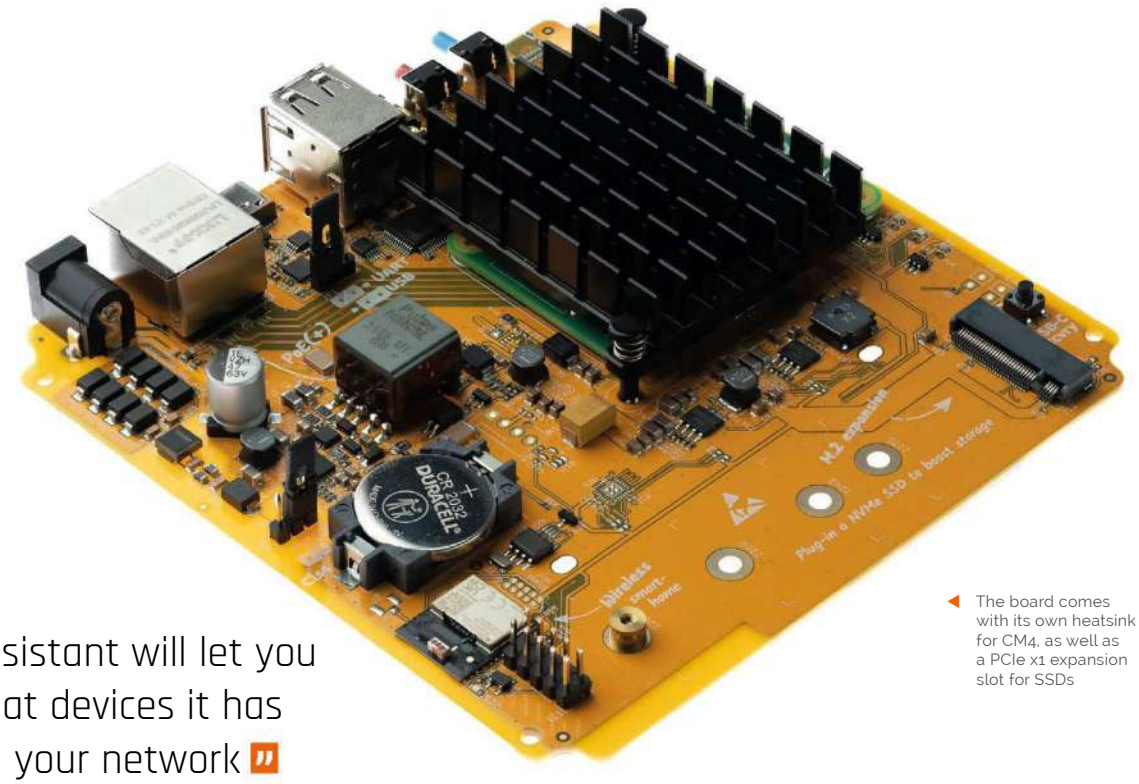
Setting up the device is very simple, although depends slightly on the version you get. The kit that comes with a Compute Module 4 already installed is basically ready to go, just requiring you to get it set up for your network and IoT add-ons.

If you're adding the Compute Module, you just need to take the case off using the hand-tightened screws (no screwdrivers involved) and firmly click CM4 in. You can also add an M.2 SSD at this stage too if you want the extra storage. After that, it's just installing the OS via a USB stick (set up with Raspberry Pi Imager), and then doing the usual network setup. The longest part of setting up this way is just letting all the software install – in about 30 minutes we were all ready to go.

Verdict

Extremely easy to set up and use, it's a very compact and practical box that really helps power your home

10/10



The board comes with its own heatsink for CM4, as well as a PCIe x1 expansion slot for SSDs

> Home Assistant will let you know what devices it has found on your network

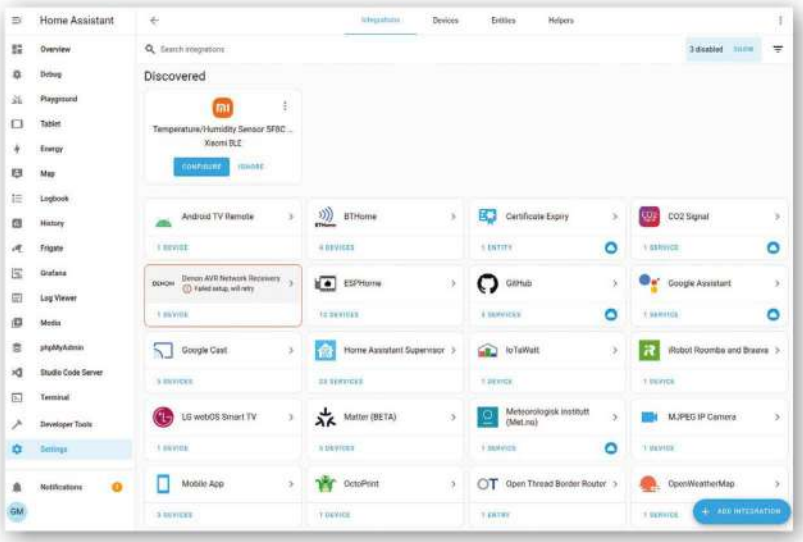
Automated automation

During the last stages of set up, Home Assistant will let you know what devices it has found on your network. It has access to a huge range of services and standards as you'd expect, and puts any found devices on your dashboard for you to configure further. This is accessible via a smartphone app or your browser, and there's an astonishing level of customisability in the software. From the use of general active triggers for actions (including voice control) to different passive 'scenes' that will, for example, automatically lower the lights when watching TV.

As this is positioned as a big hub on your network, you can also add plenty of other software services to it – such as Plex, Sonos, TOR, and a whole variety of extensions that you can easily add from the browser. The SSD slot will come in handy if you use the media server applications.

It's a fantastic piece of kit that we found very easy to use without sacrificing any of the nitty-gritty home automation tools you'd want to use to truly customise your home – to the point that it's inspired us to expand the amount of automated devices in our home.

The dashboard surfaces a lot of information so that you can perfectly tweak your home



10 amazing: robot projects

Automatons, rovers, and other robotic friends made with Raspberry Pi or Pico

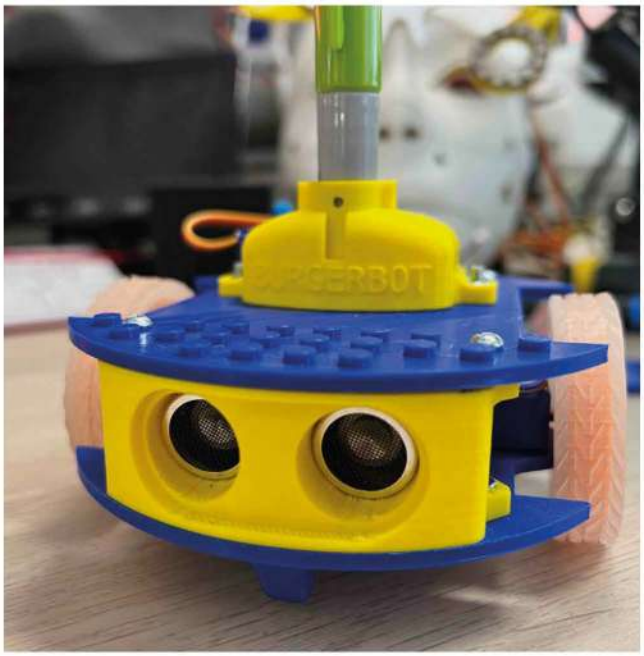
One of our favourite things to make with Raspberry Pi is a cool robot. Whether you're making them fight at Pi Wars or racing them in Formula Pi, it's an incredible way to learn about coding and making. We've put together a list of very different robots to give you some inspiration

▼ BurgerBot

Inedible round robot

This Pico-powered robot is small and compact, and a great base for expanding and building bigger and more complex robots

magpi.cc/burgerbot



▶ PicoTico

Tic-tac-toe-bot

Would you like to play a game of tic-tac-toe? This robot will play with you, but unlike WOPR (from 1983's *WarGames*) it's a bit more physical

magpi.cc/picotico

◀ NE-5

Who's Johnny?

This compact robot inspired by *Short Circuit*'s Johnny 5 is also powered by Raspberry Pi 5. It's a very advanced robot with stereoscopic vision

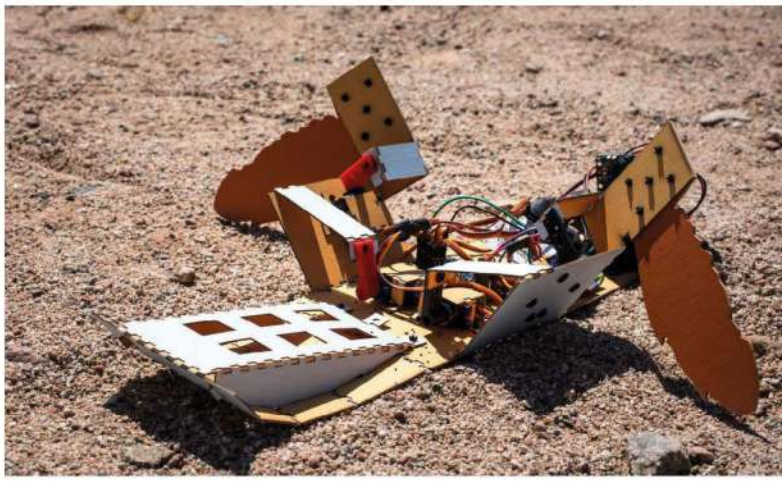
magpi.cc/ne5

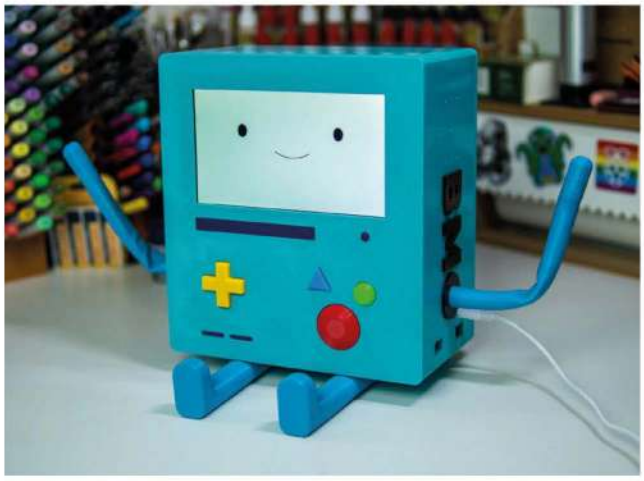
▼ C-Turtle

Landmine clearing robot

This clever and very cheap robot is great for clearing minefields, which hopefully is not something you'll have much need for. If you do, though, know it's made of cardboard and does move like a turtle.

magpi.cc/cturtle





▲ BMOctoprint

3D printing buddy

This helpful robot keeps an eye on your 3D printer and looks very cute while doing it. It's also 3D printed itself!

magpi.cc/bmoctoprint

◀ AI-suke the robot

Remote greeter

This robot uses facial recognition to greet people at the door, and was created during COVID lockdowns. It alerts the home owner when someone arrives, and even talks to them

magpi.cc/aisuke

▼ Open Weed Locator

Farming robot

Weeding can be a long task that requires a sharp eye and precision, a perfect use case for machine learning on a robot in a big field, which uses computer vision to detect and then pick weeds

magpi.cc/openweed

▲ CUBOTino

Toy puzzle solver

The theory behind Rubiks cube solving is fascinating, and so is the sport of doing it with speed. While this robot won't be beating any world records, it's very fun to watch.

magpi.cc/cubotino

▼ Olga The Fortune Teller

Clairvoyant AI

This fortune teller uses ChatGPT to predict your fate. It's probably not going to be correct, but it's also about as accurate as star sign horoscopes, so your mileage may vary

magpi.cc/olga

▼ K-9

Famous robot dog

This full-size replica of the classic Doctor Who prop recreates the original and adds several features the older version didn't have, such as touch sensors

magpi.cc/k9replica

Learn Visual Studio Code with Raspberry Pi

Get to grips with Microsoft's popular code editor with these resources. By **Phil King**

Introductory videos

AUTHOR

Microsoft

Price:
Free

magpi.cc/vscintrovid

Microsoft's Visual Studio Code is a lightweight but powerful cross-platform code editor that offers great debugging tools and a huge system of extensions. It can be used with a host of programming languages, including C/C++, C#, Python, Java, JavaScript, Markdown, and Node.js.

Best of all, Visual Studio Code is available for Raspberry Pi and can be easily installed using the Recommended Software tool (found in Menu > Preferences). You can then launch it from the Programming submenu.

The best place to start learning about Visual Studio Code is via Microsoft's own series of introductory video tutorials. The first one walks you through setting up Visual Studio Code and gives an overview of its basic features – all in just seven minutes. You can then move on to Code Editing, Productivity Tips, Personalize (using themes), Extensions, Debugging, Version Control, and Customize (settings and keyboard shortcuts).

Online courses

Enrol in a Visual Studio Code web course today

VSCODE POWER USER

This paid-for course comprises 65 videos in which the engaging Ahmad Awais delves deep into VS Code and offers time-saving tricks.

▶ vscode.pro

MASTERING VISUAL STUDIO CODE

One of several Udemy courses on VS Code, it aims to boost your productivity. Modules cover several coding languages and many useful topics.

▶ magpi.cc/vscudemy

VISUAL STUDIO CODE

Pluralsight's course by John Papa is highly rated and covers the basics along with advanced user tricks like multi-cursor and task automation.

▶ magpi.cc/vscpapa

Visual Studio Code Can Do That?

AUTHOR

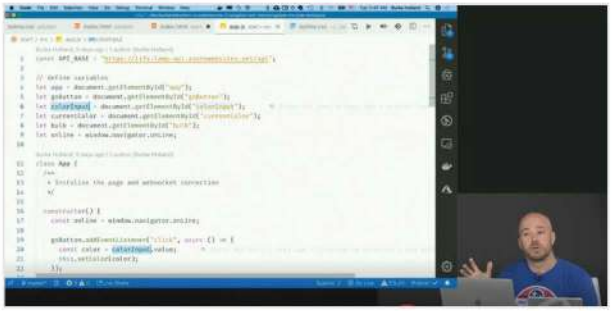
Front End Masters

Price: Free

magpi.cc/vscodecando

By opening a free Front End Masters account, you can gain access to five video courses, such as this one on Visual Studio Code presented by Burke Holland.

Aimed at increasing your speed and productivity, it promises to examine the code editor from top to bottom. Along with the fundamentals, it shows you how to turn Visual Studio Code into a full IDE (integrated development environment) to refactor and debug code, work with databases, and more.



Following an introduction and tour of the editor, course sections (each comprising multiple videos) include customising the editor, productivity tricks, navigation and refactoring, debugging, Docker, remote development, working with data, and Git. In all, the course comprises over 3.5 hours of video. Resources and downloads can be found in the GitHub repo and GitBook.

Extension Marketplace Guide

AUTHOR

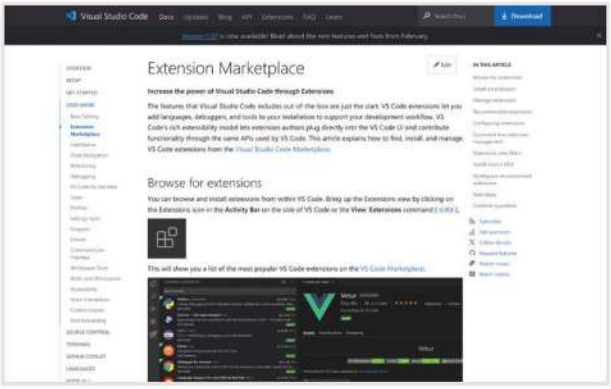
Microsoft

Price: Free

magpi.cc/vscextensions

One of the best features of Visual Studio Code is that you can customise just about everything in it by adding extensions.

You'll certainly be spoilt for choice as there are over 30,000 extensions available, offering a huge array of potential extra features for the code editor. These range from testing and security tools to productivity boosters (such as code analysers and cleaners) and collaboration extensions such as LiveShare. You can even create your own extensions if you know how.



The official Extension Marketplace documentation shows how to find, install, and manage VS Code extensions from the Visual Studio Code Marketplace. In the editor, selecting View > Extensions shows you the most popular ones, each with a brief description and user rating. Try a few out to see if they can help you.

Useful websites

Some sites to visit for helpful info

KEY BINDINGS

This useful documentation page shows you the default keyboard shortcuts for Visual Studio Code, and also how to change them.

▶ magpi.cc/vskeys

MARKETPLACE

Browse the thousands of extensions available for Visual Studio Code, the vast majority of them free, with helpful filter and sort options.

▶ magpi.cc/vscmarket

AWESOME VSCODE

Want even more resources? Viatsko's GitHub repo has a curated (and long) list of useful links, helpfully arranged into categories.

▶ magpi.cc/awesomevscode

André Costa

Had trouble finding a Raspberry Pi during the supply chain problems? André and his rpilocator had your back

> Name **André Costa** | > Occupation **Tech support**
> Community role **Stock tracker** | > URL **rpilocator.com**

For a while it was a bit difficult to get a new Raspberry Pi. With the global supply chain still recovering from COVID, people who wanted a Raspberry Pi had to wait patiently for Approved Resellers to get new stock – or pay exorbitant prices second hand. This is where André Costa came to the rescue with rpilocator.

"Sometime around November of 2021, I started on a project making a carrier board for the CM4 and I needed some units to test my prototypes," Andre tells us. "That's when I noticed the availability of Raspberry Pi computers in general was low. Initially, I registered for stock notifications at a couple of authorised resellers. I got a couple of email notifications a few days in a row, but they were always sold out before I could go to the seller's website."

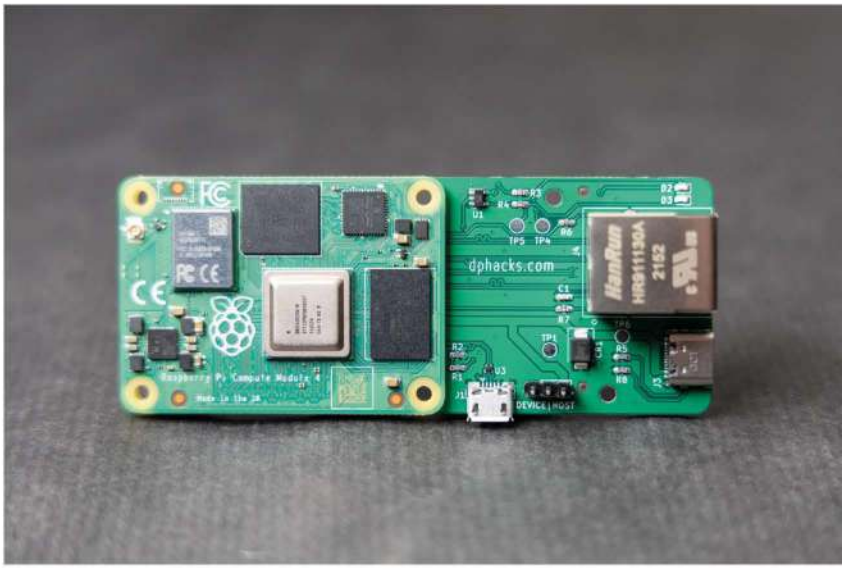
The service we now know as rpilocator started off as cm4locator, with André coding it during a couple of days off. Initially it was private, and within a couple of days it had helped him locate – and buy – a Raspberry Pi Zero 2 W. Surprised at how easy it was, he decided to make a public version.

"I registered the domain on January 26 and spent a couple of hours a day after work adding more listings to be tracked," André says. "On January 29 I pushed the website to a cloud service. Rpilocator was officially live but no one knew about it. I waited a day to make sure everything was working and I sent an email to Lee from leepspvideo and Jeff Geerling."

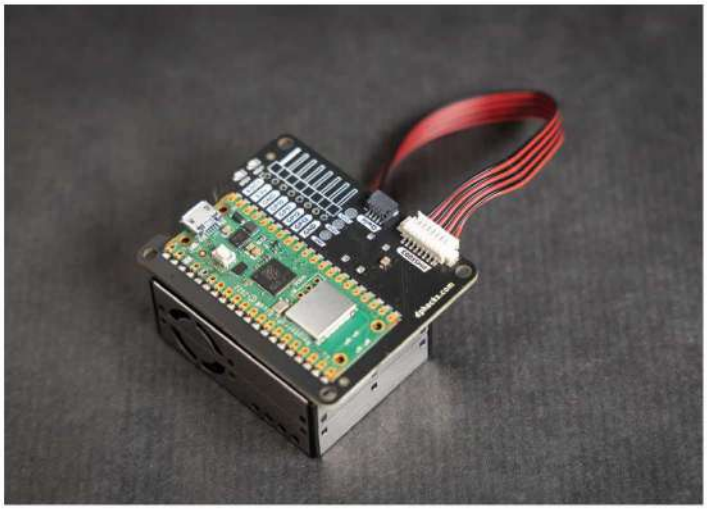
From there, word spread, and rpilocator became the best way to find a Raspberry Pi.

What is your history with making?

I grew up in a household where both my parents were very handy at making things. My dad is an engineer who loves tinkering



▶ The project that spurred rpilocator is an Ethernet board for Compute Module 4



▼ This Raspberry Pi Pico-powered air quality sensor helped André and his family when Canada wildfires caused issues as far away as Brazil

> My pie-in-the-sky project would be something that combines scientific knowledge and artistic expression

with everything. I remember one of the first projects we worked on together was a crystal radio receiver – the classic kid introduction to making project.

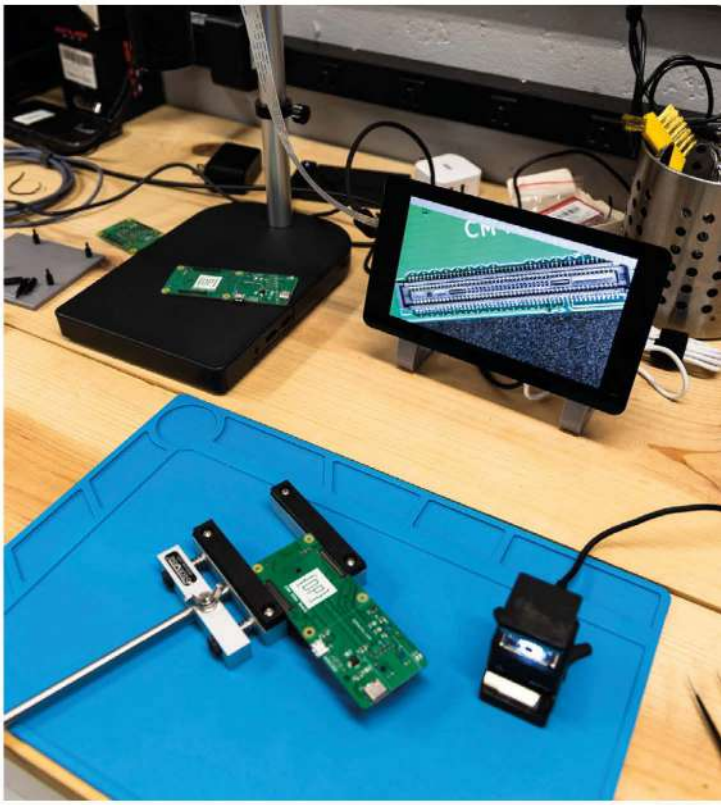
I grew up in Brazil and access to maker kits wasn't as widely available as in the US or the UK. This was in the '80s and the '90s. There were some STEM magazines available in English and that was one of the reasons why I wanted to learn English.

My mom started sewing when she was young and she's an experienced seamstress. Seeing a flat sheet of fabric turn into a 3D object was so inspiring to me. I've played with and created things with textiles my whole life.

What projects have you made with Raspberry Pi?

I've used Raspberry Pi in quite a few projects over the years ranging from a network of sensors to home automation and camera applications.

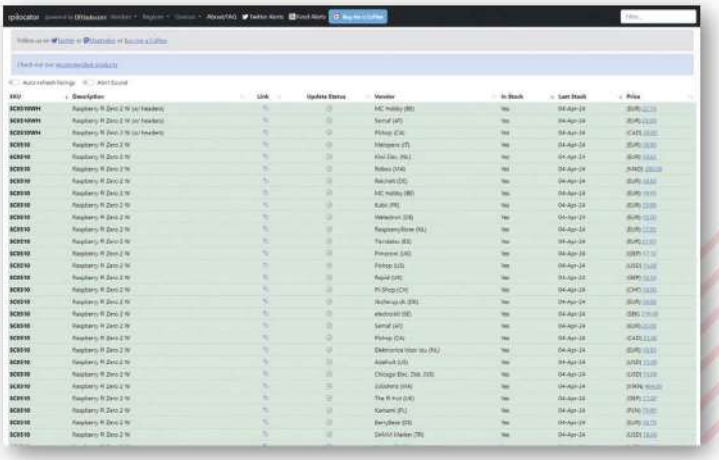
Last year, there were many large wildfires in Canada. A lot of the smoke drifted into where we live. The air quality outside was unhealthy for long periods of time. My daughter hadn't turned one yet and I wanted to make sure the air quality inside our home was healthy for her. I created a PCB that hosts a Pico W and makes it easy to plug an air quality sensor and I2C devices to the Pico W. It runs an open-source CircuitPython firmware that calculates the Air Quality Index inside our home and sends sensor information through MQTT.



▲ A Raspberry Pi Camera Module zooms in on André's soldering efforts

What is your dream project?

I find the intersection between science and art so intriguing. My pie-in-the-sky project would be something that combines scientific knowledge and artistic expression. I don't know exactly what it would be. Maybe something that would spark kids' interest in learning more about science and art.



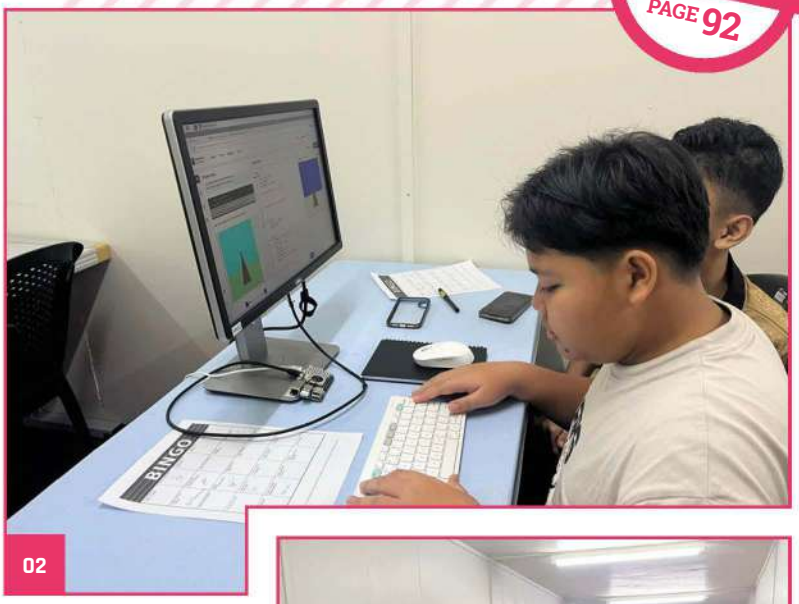
◀ There's a lot of green on rpilocator now that supply chain issues are mostly resolved around the world

Events in pictures: Program Komuniti Budak Coding

Community and official events in the wild

FIND OUT ABOUT NEXT MONTH'S EVENTS AND POP-UPS ON PAGE 92

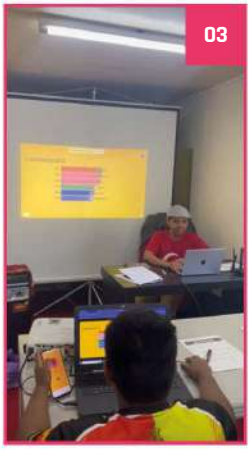
This Malaysian coding event for kids kicked off with learning about the terminal on Raspberry Pi OS, with Python planned for the next session. Apparently the kids stormed through with the terminal and were able to start learning about simple Python games.



02

01. The classroom was small, but the brains on the kids participating definitely were not
02. Making use of Raspberry Pi Foundation projects to learn Python with game programming
03. A bingo game was done to break the ice early in the sessions
04. "We hope, with this small effort, we are able to produce Malaysians who invent new technologies," say the organisers

01



03

04

Pico Throttle

With recycled cardboard and a potentiometer, you can get analogue flight control

▲ After eating some chocolate, you just want to fly around the sky in some virtual planes

▲ It's held together tape, hot glue, and we assume a few prayers to the Easter bunny

Leo emailed us early in April to show off his throttle lever that he'd made out of Easter egg boxes (and a lot of hot glue). It's an ingeniously simple construction – a Raspberry Pi Pico measures a potentiometer, which is attached to a cardboard handle on a little rod so it can turn. Using HID code with CircuitPython, he can then send an analogue signal to his PC as a control axis – perfect for flight sims or *Elite Dangerous*. It also features a digital button on the back of the throttle to set maximum reverse thrust for when you move the lever beyond the idle position.

"I've often had the occasional flight on my grandad's copy of *Flight Simulator X* when I visit him, but I purchased *X-Plane 12* after Christmas, so I've been using it regularly for about three months." Leo tells us about what inspired him. "[The throttle is] reasonably rigid. There's some flex in the lever side-to-side, so the plate that pushes the reverser button has to be a bit wider, but the box and paper-tube pivot is solid enough. It is, however, very light and likes to move around the desk."

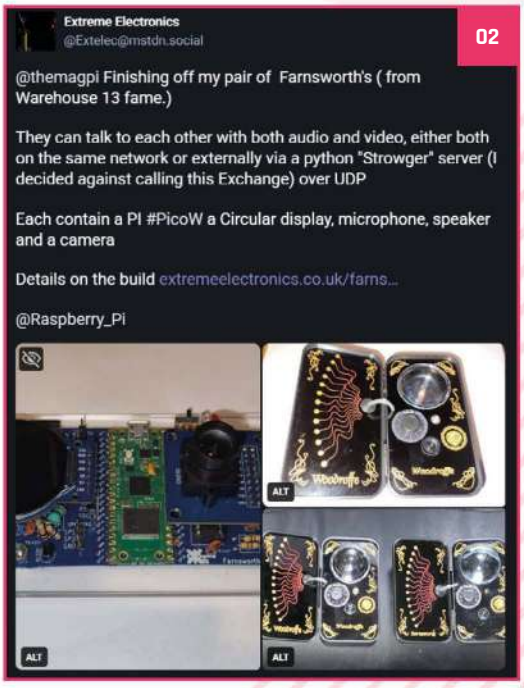
▶ The full setup includes a Raspberry Pi displaying data exchange plugin SimVim for more interactivity

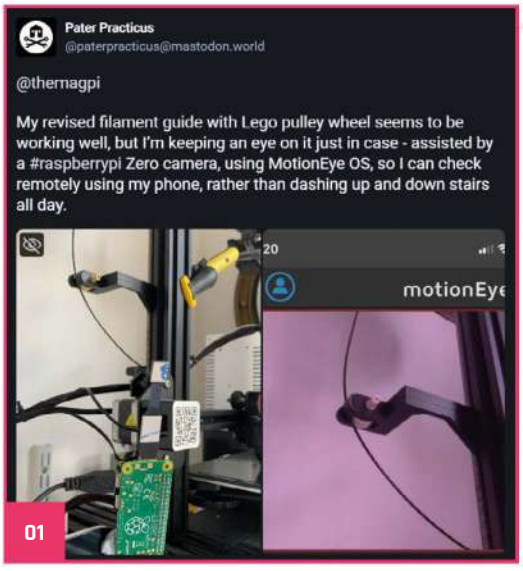
MagPi Monday

Amazing projects direct from social media!

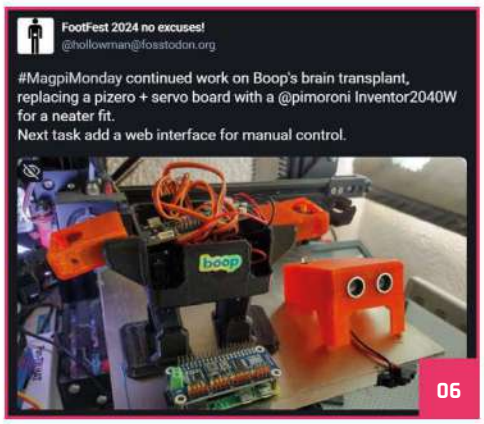
Every Monday we ask the question: have you made something with a Raspberry Pi over the weekend? Every Monday, our followers send us amazing photos and videos of the things they've made.

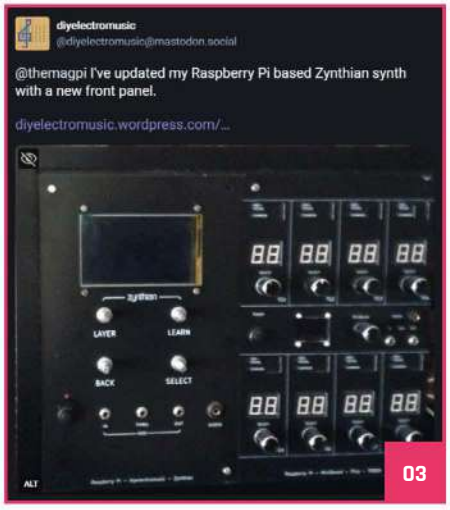
Here's a selection of some of the awesome things we got sent this month. Remember to follow along at the hashtag #MagPiMonday!

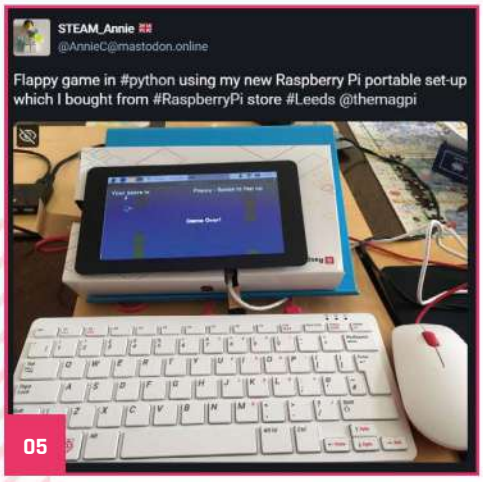


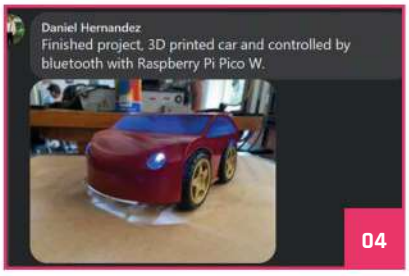


01. People often use 3D printed items to modify their 3D printers, but we like to see some LEGO in use
02. This is based on a fictional 1929 wireless communicator, and looks very cool
03. Music updates are still continuing!
04. This is great, we love how cheap and simple it must have been to build
05. We love a good portable Raspberry Pi – we should do another feature for Raspberry Pi 5
06. We wish Boop good luck with its new brain


FootFest 2024 no excuses!
@hollowman@fosstodon.org
#MagpiMonday continued work on Boop's brain transplant, replacing a pizero + servo board with a @pimoroni Inventor2040W for a neater fit.
Next task add a web interface for manual control.
boop
06


diyelectromusic
@diyelectromusic@mastodon.social
@themagpi I've updated my Raspberry Pi based Zynthian synth with a new front panel.
diyelectromusic.wordpress.com/...
LAYER
LEARN
BACK
SELECT
03


STEAM_Annie
@AnnieC@mastodon.online
Flappy game in #python using my new Raspberry Pi portable set-up which I bought from #RaspberryPi store #Leeds @themagpi
05


Daniel Hernandez
Finished project, 3D printed car and controlled by bluetooth with Raspberry Pi Pico W.
04

Your Letters

Weather forecasting

I am attempting to complete the 'Build a weather station' project by Phil King described on page 132 of the 2024 Official Raspberry Pi Handbook. I am having a problem which is described below.

I am using a Raspberry Pi 4 with Bullseye. Per the instructions, I have set up Raspberry Pi, mounted the HAT, installed the software and rebooted Raspberry Pi. However, when I perform step four, 'Initial testing', I get the following error message:

```
$ cd weatherhat-python/examples

$ python weather.py

Traceback (most recent call last):

  File "/home/mbliz/weatherhat-python/examples/weather.
py", line 738, in <module>

    main()

  File "/home/mbliz/weatherhat-python/examples/weather.
py", line 698, in main

    display = ST7789.ST7789(

  File "/home/mbliz/.local/lib/python3.9/site-packages/
ST7789/__init__.py", line 123, in __init__

    self._spi = spidev.SpiDev(port, cs)

FileNotFoundError: [Errno 2] No such file or directory
```

I am a Raspberry Pi neophyte, so I am unclear what the error message is referring to, much less how to troubleshoot it. I would be very grateful if you could give me some guidance on how to proceed.

Mike via email

With the ever-changing nature of technology, we discovered while helping Mike that to get this project working that you'll now need to install the program slightly differently. The original version of this tutorial has you build the software, however it's now available in the Python PIP repository.

If you ever get stuck in a coding project like this, we always suggest checking the original website (in this case **magpi.cc/weatherhatgit**), and you can also drop us an email at **magpi@raspberrypi.com**. We will do our best to point you in the right direction.

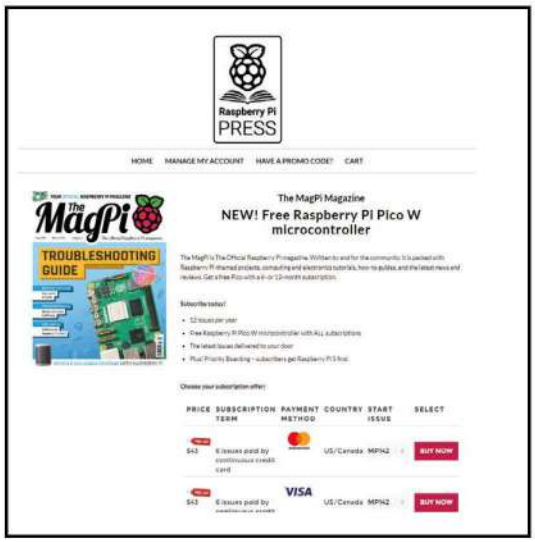
▲ The Weather HAT project is well worth your time – it's useful and a great way to learn about meteorology

US subscriptions

Hi, does the "Save 35% off the cover price with a subscription to The MagPi magazine" for UK subscribers apply to me if I live in the USA if I want to get a 12-month subscription?

Austin via email

Our subscription offers are slightly different depending on where you live – the discount is based on how much you'd spend buying each issue individually versus the cost of the subscription as a whole, so the rate on the US subscriptions is slightly different. You can find out about all our subscription offers here: **magpi.cc/subscribe**



▲ Our current subscriptions also offer you a free Raspberry Pi Pico W on some tiers!

Contact us!

- Mastodon **magpi.cc/mastodon**
- Threads **@themagpimag**
- Facebook **magpi.cc/facebook**
- Email **magpi@raspberrypi.com**
- Online **forums.raspberrypi.com**

Community Events Calendar

Find out what community-organised Raspberry Pi-themed events are happening near you...

01. Pi Force

Friday 3 May
Lincoln Corners Pakistan, Karachi, Pakistan
magpi.cc/piforce141

Come join us for a day of tinkering, hacking and learning – we will be having multiple sessions from basic electronics and soldering, to Arduino, then moving to Raspberry Pi as a desktop PC, hacking PC, and robotics PC.

02. Mayday Pi Jam Leicester

Monday 6 May
Leicester Hackspace Unit 40, Leicester, UK
magpi.cc/mayday141

Leicester Hackspace is holding another Pi Jam on Bank Holiday Monday, May 6, between noon and 4pm. The focus will be on using the camera for posture detection, astronomy and as a nature cam. Along with robots, of course, Raspberry Pi Pico and other Raspberry Pi projects.

03. Riverside Raspberry Pi Meetup

Monday 13 May
3600 Lime Street, Riverside, CA, USA
magpi.cc/rrpm141

The purpose of Riverside Raspberry is to share knowledge related to Raspberry Pi hardware in particular, and to promote interest in tech development in California's Inland Empire in general. The group is currently meeting on the second Monday evening of every month.

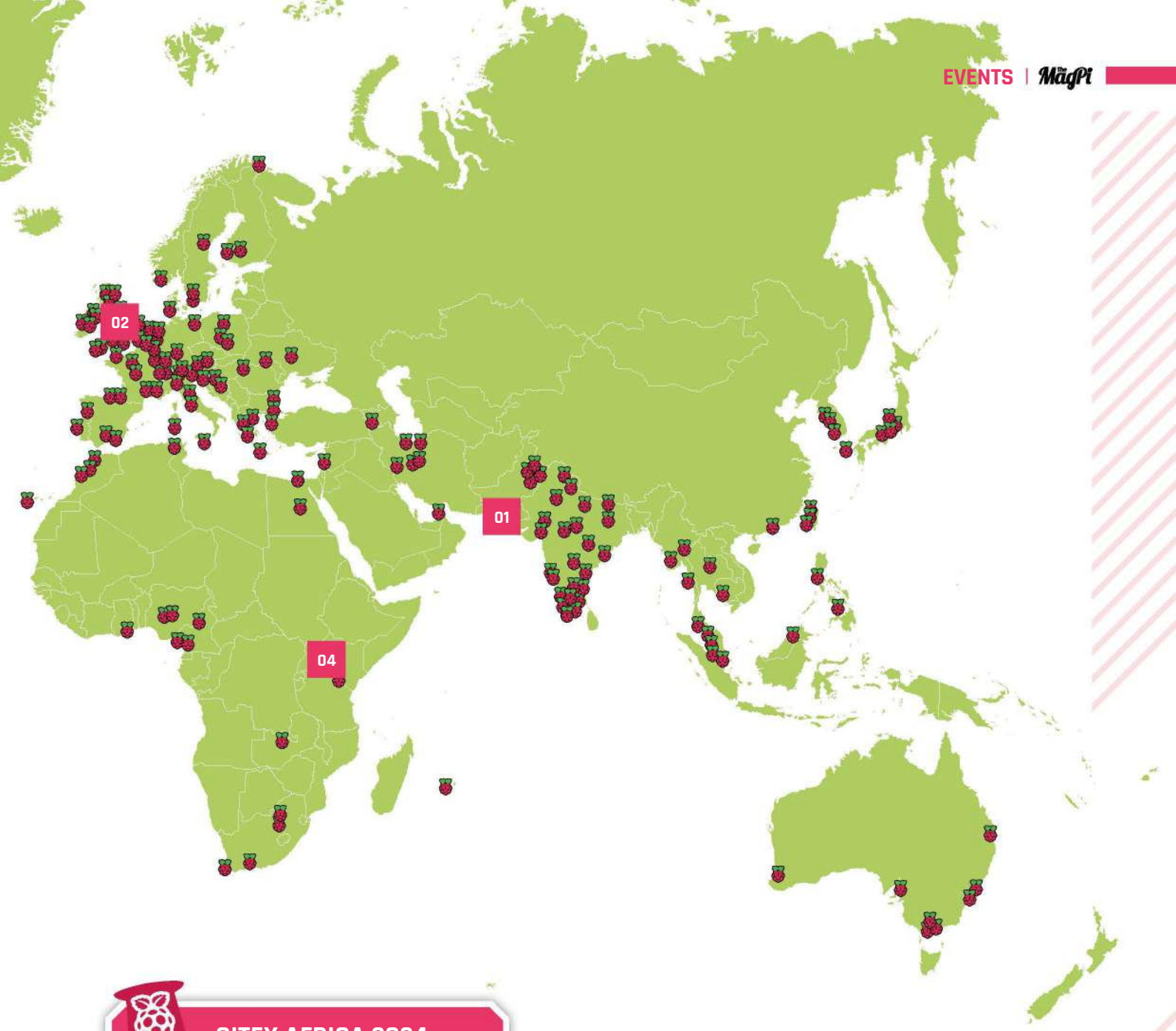
04. Kisii Raspberry Pi Jam

Saturday 18 May
Kereri Girls School, Kisii, Kenya
magpi.cc/kisii141

Kisii Raspberry Pi Jam: igniting young minds through tech exploration!

Calling all tech enthusiasts in Kisii! Join us for an exciting Raspberry Pi Jam at Kereri Girls School, a premier public girls' school known for its academic excellence. This event is designed to introduce students to the wonders of technology through hands-on workshops and exploration with Raspberry Pi kits.

FULL CALENDAR
Get a full list of upcoming community events here:
magpi.cc/events



Gitex Africa 2024

- Where Place Bab Jdid, Marrakesh, Morocco
- When Wednesday 29 May to Friday 31 May

The Raspberry Pi team is delighted to be visiting Marrakesh, Morocco for our first visit to Africa's biggest tech and startup show: Gitex Africa 2024. There you'll be able to meet our team and experience the full range of our technology, including Raspberry Pi Pico, RP2040-based solutions, Compute Modules and Raspberry Pi single-board computers as well as cameras and our range of high-quality accessories.

magpi.cc/gitex24

WIN AN ED-HMI3020 TOUCHSCREEN

This rugged display can safely and securely house a Raspberry Pi 5, and features a ten-inch touchscreen with a resolution of 1200×800. The screen is designed for industrial settings, but also works great as a portable kiosk in your home or office.

Head here to enter: magpi.cc/win | Learn more: magpi.cc/edhmi3020

Terms & Conditions

Competition opens on **24 April** and closes on **30 May 2024**. Prize is offered to participants worldwide aged 13 or over, except employees of Raspberry Pi Ltd, the prize supplier, their families, or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram, Facebook, Twitter or any other companies used to promote the service.

HackSpace

TECHNOLOGY IN YOUR HANDS

THE MAGAZINE FOR THE MODERN MAKER

SUBSCRIBE AND SAVE UP TO 35% on the cover price

ISSUE **#78**

OUT NOW

hsmag.cc

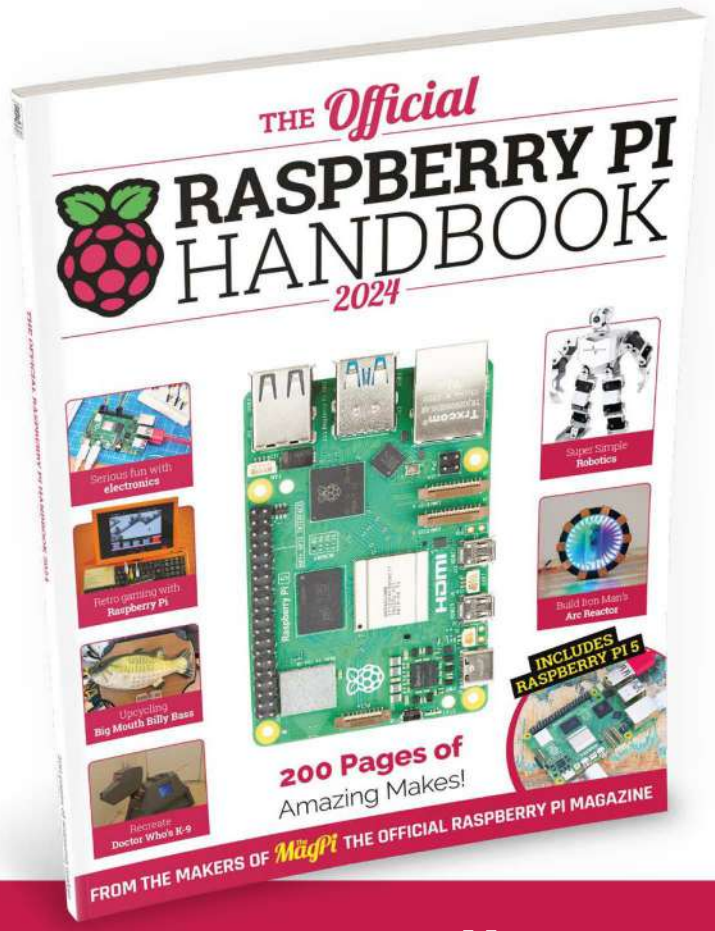
Available on the App Store

GET IT ON Google Play

THE Official RASPBERRY PI HANDBOOK 2024

200 PAGES OF RASPBERRY PI

- QuickStart guide to setting up your Raspberry Pi computer
- Updated with Raspberry Pi Pico and all the latest kit
- The very best projects built by your Raspberry Pi community
- Discover incredible kit and tutorials for your projects



Buy online: magpi.cc/store

Raspberry Pi 5 Media Player

Build a next-gen media player with Raspberry Pi 5 and Argon ONE V3 Case

The Magpi **#142**
On sale **30 May**

Plus!

Save a ZX Spectrum with Pico

Python data structures

Summer projects

DON'T MISS OUT! **magpi.cc/subscribe**

MASTODON	magpi.cc/mastodon
THREADS	@themagpimag
FACEBOOK	magpi.cc/facebook
EMAIL	magpi@raspberrypi.com
ONLINE	forums.raspberrypi.com

EDITORIAL

Editor
Lucy Hattersley
lucy@raspberrypi.com

Features Editor
Rob Zwetsloot
rob@raspberrypi.com

Sub Editor
Ian Evenden

ADVERTISING

Charlotte Milligan
charlotte.milligan@raspberrypi.com
+44 (0)7725 368887

DESIGN

Head of Design
Jack Willis

Designers
Sara Parodi, Natalie Turner

Illustrator
Sam Alder

Photographer
Brian O'Halloran

CONTRIBUTORS

David Crookes, PJ Evans, Gareth Halfacree, Rosie Hattersley, Phil King, Rob Miles, K.G. Orphanides

PUBLISHING

Publishing Director
Brian Jepson
brian.jepson@raspberrypi.com

Director of Communications
Liz Upton

CEO
Eben Upton

DISTRIBUTION

Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

SUBSCRIPTIONS

Unit 6 The Enterprise Centre
Kelvin Lane, Manor Royal,
Crawley, West Sussex, RH10 9PE
+44 (0)1293 312193
magpi.cc/subscribe
magpi@subscriptionhelpline.co.uk

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

The MagPi magazine is published by Raspberry Pi Ltd, 194 Cambridge Science Park, Milton Road, Cambridge, England, CB4 0AB. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products, or services referred to or advertised in the magazine. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0).
ISSN: 2051-9982.

Space

What to do when you suddenly have a desk in the office, wonders **Rob Zwetsloot**

I've probably mentioned here before, or it may be just common knowledge because I tend to reveal too much personal information, but the editorial team on *The MagPi* and *HackSpace* magazines generally work from home. We all visit the main office once a month, but editor Lucy heads in every week because she's just that dedicated (and lives a lot closer than the rest of us [it's the free coffee machine – Ed!]). Anyway, Raspberry Pi recently moved to a new office, and for the first time since joining Raspberry Pi, I now have a desk!

This has caused me a little bit of a quandary though. My desk in the office is my home away from home and I want to make it feel comfy and welcoming when I visit. However, I'm not there that often, so I can't bring in stuff I need at home, where I do most of my work.

Sparse

Currently I've brought in an original Raspberry Pi Zero from the cover of an issue 40, still in its blister pack, a 3D POP camera, some jingle bells I found, and a little ceramic hedgehog with a fake succulent plant growing out of it. It's also got some prizes there for future competitions but they'll come and go so that's a bit different. The benefit of bringing stuff in also means I won't have it filling up my home any more, but don't tell the boss that.

I've been thinking about having little Raspberry Pi and/or Pico-powered devices to keep there – I have threatened to make a mini set of two drums and a cymbal so I can make a rimshot machine to punctuate my steady stream of bad jokes – and as we have a fancy new maker lab here with lots of toys – er, I mean serious tools – I feel I should make use of that while I'm visiting. Send me suggestions for simple little machines to the usual places!

> I have threatened to make a mini set of two drums and a cymbal so I can make a rimshot machine to punctuate my steady stream of bad jokes

Something to call your own

This is clearly a fairly unique and low-level problem, but I wanted to bring it up because it is important to be able to have a space to work/indulge in hobbies at while being comfortable. I have the luxury of being able to personalise my working space at home as well as my hobby space, which is full of boom arms holding cameras, lights and microphones though. I may have chosen the wrong hobby.

It's taken me a while to get to this stage, so don't get frustrated if it takes a while for you to get a space you can feel comfy in. Just like learning any skill, it can take practice as well as some trial and error. Also, money.

I now need to figure out the work policy on how many hedgehog plushes I can keep on my desk.

AUTHOR

Rob Zwetsloot

Rob last had a desk in an office nine years ago, where he kept old Power Ranger toys and a foam sword from a tech conference he attended at Walt Disney World

magpi.cc

HiPi.io

HIGHPI PRO

The new case from the HiPi.io team

- Rapid tool-free assembly and disassembly
- Large internal volume for HATs
- Compatible with Pi 2/3/4
- Multiple lid options
- Passive & Active Cooling options
- Secure microSD card cover
- VESA mount support
- Molding-configurable output ports customizable for volume orders
- Printed logo for your branding

Available at these great Pi stores:

PiShop.us PiShop.ca PiShop.mx ThePiHut Pi-Shop.ch Welectron.

Contact your favorite Pi store if it's not listed here

PiKVM

Remote control redefined

Manage your servers or PCs remotely!

POWERED BY Raspberry Pi

PiKVM V4 Mini

Small, cost-effective, and powerful!

- Power consumption in idle mode: just 2.67 Watts!
- Transfer your mouse and keyboard actions
- Access to all configuration settings like UEFI/BIOS
- Capture video signal up to 1920x1200@60 Hz
- Take full control of a remote PC's power

PiKVM V4 Plus

The most feature-rich edition

- More connectivity
- Extra storage via internal USB 3.0
- Upgraded powering options
- More physical security features
- Extra HDMI output
- Advanced cooling solution

A cost-effective solution for data-centers, IT departments or remote machines!

Available at the main Raspberry Pi resellers

core ELECTRONICS · PiShop.ca · RPishop.cz · Welectron. · elektorstore · PIMORONI

Pi-Shop.ch · The PiHut · PiShop.us · CloudFree · KUBii · Cytron Marketplace

ROBU.IN Your ideas, Our Parts · NETTOP COMPUTERS & ELECTRONICS · netsket mall · DIREKTRONIK · RaspberryPi.dk

HiPi.io

No reseller in your country?
Check shop.hipi.io (import fees might apply).

List of official resellers by country: